

glossaries-extra.sty v1.30: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2018-04-25

Abstract

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

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

Contents

1	Main Package Code (glossaries-extra.sty)	5
1.1	Package Initialisation and Options	5
1.2	Extra Utilities	27
1.3	Modifications to Commands Provided by glossaries	35
1.3.1	Existence Checks	39
1.3.2	Document Definitions	47
1.3.3	Existing Glossary Style Modifications	52
1.3.4	Entry Formatting, Hyperlinks and Indexing	56
1.3.5	Entry Counting	93
1.3.6	Acronym Modifications	107
1.3.7	Indexing and Displaying Glossaries	109
1.4	Link Counting	145
1.5	Integration with glossaries-accsupp	147
1.6	Categories	157
1.7	Abbreviations	183
1.7.1	Abbreviation Styles Setup	203
1.7.2	Predefined Styles (Default Font)	206
1.7.3	Predefined Styles (Small Capitals)	223
1.7.4	Predefined Styles (Fake Small Capitals)	237
1.7.5	Predefined Styles (Emphasized)	251
1.7.6	Predefined Styles (User Parentheses Hook)	273
1.7.7	Predefined Styles (Hyphen)	282
1.7.8	Predefined Styles (No Short on First Use)	296
1.8	Using Entries in Headings	299
1.9	Multi-Lingual Support	318
1.10	glossaries-extra-bib2gls.sty	319
2	Style Adjustments (glossaries-extra-stylemods.sty)	355
2.1	Package Initialisation	355
2.2	List-Like Styles	356
2.3	Longtable Styles	359
2.4	Long Ragged Styles	361
2.5	Supertabular Styles	363
2.6	Super Ragged Styles	365
2.7	Inline Style	367
2.8	Tree Styles	367
2.9	Multicolumn Styles	384

3 bookindex style (glossary-bookindex.sty)	390
3.1 Package Initialisation and Options	390
Glossary	396
Change History	397
Index	415

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2018/04/25 v1.30 (NLCT)]
```

Requires xkeyval to define package options.

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

Requires etoolbox package.

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

Has glossaries already been loaded?

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

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

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

Not already loaded, so pass options to glossaries.

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

Set the defaults.

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

Declare package options.

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

`\f@for@gl@sentries`

```

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

```

`\f@for@gl@sentries`

```

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

```

```

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

63\define@choicekey{glossaries-extra.sty}{undefaction}%
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.

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

```

83 \newcommand*{\@glsxtr@do@record@wrglossary}[1]{%
84   \begin{group}
85     \ifKV@gls@link@noindex
86     \else
87       \edef\@gls@label{\glsdetoklabel{#1}}%
88       \let\gls@label\@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
110          Increment associated counter.
111          \glsxtr@inc@wrglossaryctr{#1}%
112          \@@do@@wrglossary
113        \fi
114      }%
115    \fi
116  \end{group}

```

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

```

116 \newcommand*{\@glsxtr@do@alsoindex@wrglossary}[1]{%
117   \glsxtr@@do@wrglossary{#1}%
118   \@glsxtr@do@record@wrglossary{#1}%
119 }

```

`\@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).

```
120 \newcommand*{\@@glsxtr@record}[3]{%
121   \ifglsentryexists{#2}{}%
122   {%
123     \@@glsxtrwrglossmark
124     \begingroup
```

Save the label in case it's needed.

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

Entry hasn't been defined, so we'll have to assume it's \glscounter by default.

```
131     \let\@gls@counter\glscounter
```

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

```
132     \@gls@setdefault@glslink@opts
```

Implement any pre-key settings.

```
133     \csuse{\@glsxtr@#3@prekeys}%
```

Assign keys.

```
134     \setkeys{#3}{#1}%
```

Implement any post-key settings.

```
135     \csuse{\@glsxtr@#3@postkeys}%
```

Increment associated counter.

```
136     \glsxtr@inc@wrglossaryctr{#2}%
```

Check if noindex option has been used.

```
137     \ifKV@glslink@noindex
138     \else
139       \glswriteentry{#2}%
140       {%
```

Check if thevalue has been set.

```
141         \ifdefempty{\@glsxtr@thevalue}%
142         {%
```

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

```
143         \ifx\@glsxtr@org@theHvalue\@glsxtr@theHvalue
144         \else
145           \let\theHglsentrycounter\@glsxtr@theHvalue
146         \fi
```

Save the entry counter.

```
147         \glsxtr@saveentrycounter
```

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

```
148      \let\@@do@@wrglossary\glxtr@dorecord
149      }%
150      {%
```

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.)

```
151      \let\theglentrycounter\glxtr@thevalue
152      \let\theHglentrycounter\glxtr@theHvalue
153      \let\@@do@@wrglossary\glxtr@dorecordnodefer
154      }%
155      \ifx\glxtr@record@setting\glxtr@record@setting@alsoindex
156      \glxtr@@do@@wrglossary{#2}%
157      \else
```

No need to escape special characters.

```
158      \@@do@@wrglossary
159      \fi
160      }%
161      \fi
162      \endgroup
163      }%
164      }
```

glslink@prekeys

```
165 \newcommand{\@glxtr@glslink@prekeys}{\glslinkpresetkeys}
```

lslink@postkeys

```
166 \newcommand{\@glxtr@glslink@postkeys}{\glslinkpostsetkeys}
```

lossadd@prekeys

```
167 \newcommand{\@glxtr@glossadd@prekeys}{\glsaddpresetkeys}
```

ossadd@postkeys

```
168 \newcommand{\@glxtr@glossadd@postkeys}{\glsaddpostsetkeys}
```

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

```
169 \newcommand*\@glxtr@dorecord{%
170   \global\let\@glsrecordlocref\theglentrycounter
171   \let\@glxtr@orgprefix\@glo@counterprefix
172   \ifx\theglentrycounter\theHglentrycounter
173     \def\@glo@counterprefix{}%
174   \else
175     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
176       {\theglentrycounter}{\theHglentrycounter}}%
177     }%
178     \@do@gls@getcounterprefix
179   \fi
```

Don't protect the `\@glsrecordlocref` from premature expansion. If the counter isn't

page then it needs expanding. If the location includes `\thepage` then `\protected@write` will automatically deal with it.

```
180 \protected@write\@auxout{}\string\glsxtr@record
181   {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
182   {\@glsrecordlocref}}%
183 \@glsxtr@counterrecordhook
184 \let\@glo@counterprefix\@glsxtr@orgprefix
185 }
```

`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.

```
186 \newcommand*\@glsxtr@dorecordnodefer{%
187   \ifx\theglsentrycounter\theHglentrycounter
188     \protected@write\@auxout{}\string\glsxtr@record
189       {\@gls@label}{\@gls@counter}{\@glsnumberformat}%
190       {\theglsentrycounter}}%
191   \else
192     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
193       {\theglsentrycounter}{\theHglentrycounter}}%
194     }%
195     \@do@gls@getcounterprefix
196     \protected@write\@auxout{}\string\glsxtr@record
197       {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
198       {\theglsentrycounter}}%
199   \fi
200   \@glsxtr@counterrecordhook
201 }
```

`r@recordcounter`

```
202 \newcommand*\@glsxtr@recordcounter{%
203   \@glsxtr@noop@recordcounter
204 }
```

`p@recordcounter`

```
205 \newcommand*\@glsxtr@noop@recordcounter}[1]{%
206   \PackageError{glossaries-extra}{\string\GlsXtrRecordCounter\space
207     requires record=only or record=alsoindex package option}{}%
208 }
```

`op@recordcounter`

```
209 \newcommand*\@glsxtr@op@recordcounter}[1]{%
210   \eappto\@glsxtr@counterrecordhook{\noexpand\@glsxtr@docounterrecord{#1}}%
211 }
```

`lsxtr@recordsee` Deal with `\glssee` in record mode. (This doesn't increment the associated counter.)

```

212 \newcommand*{\@glsxtr@recordsee}[2]{%
213   \@@glsxtrwrglossmark
214   \def\@gls@xref{#2}%
215   \@onelevel@sanitize\@gls@xref
216   \protected@write\@auxout{}\string\glsxtr@recordsee{#1}{\@gls@xref}}%
217 }

```

srtglossaryunit

```

218 \newcommand{\printunsrtglossaryunit}{%
219   \print@noop@unsrtglossaryunit
220 }

```

tr@setup@record Initialise.

```

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

```

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

```

222 \newcommand*{\glsxtr@indexonly@saveentrycounter}{%
223   \ifKV@glslink@noindex
224   \else
225     \glsxtr@saveentrycounter
226   \fi
227 }

```

addloclistfield

```

228 \newcommand*{\glsxtr@addloclistfield}{%
229   \key@ifundefined{glossentry}{loclist}%
230   {%
231     \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
232     \appto\@gls@keymap{,{loclist}{loclist}}%
233     \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
234     \appto\@newglossaryentryposthook{%
235       \gls@assign@field{\@glo@label}{loclist}{\@glo@loclist}%
236     }%
237     \glssetnoexpandfield{loclist}%
238   }%
239   {%

```

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

```

240   \key@ifundefined{glossentry}{location}%
241   {%
242     \define@key{glossentry}{location}{\def\@glo@location{##1}}%
243     \appto\@gls@keymap{,{location}{location}}%
244     \appto\@newglossaryentryprehook{\def\@glo@location{}}%
245     \appto\@newglossaryentryposthook{%
246       \gls@assign@field{\@glo@label}{location}{\@glo@location}%
247     }%
248     \glssetnoexpandfield{location}%
249   }%
250   {%

```

Add a key to store the group heading.

```

251 \key@ifundefined{glossentry}{group}%
252 {%
253   \define@key{glossentry}{group}{\def\@glo@group{##1}}%
254   \appto\@gls@keymap{,{group}{group}}%
255   \appto\@newglossaryentryprehook{\def\@glo@group{}}%
256   \appto\@newglossaryentryposthook{%
257     \gls@assign@field{\@glo@label}{group}{\@glo@group}%
258   }%
259   \glssetnoexpandfield{group}%
260 }%
261 {}%
262 }

```

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

```

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

```

@alsoindex

```

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

```

@only

```

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

```

@off

```

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

```

Now define the record package option.

```

267 \define@choicekey{glossaries-extra.sty}{record}
268 [ \@glsxtr@record@setting\glsxtr@record@nr ]%
269 {off,only,alsoindex}%
270 [only]%
271 {%
272   \ifcase\glsxtr@record@nr\relax

```

Don't record.

```

273   \def\glsxtr@setup@record{%
274     \renewcommand*{\@do@seeglossary}{\@glsxtr@doseeglossary}%
275     \renewcommand*{\@glsxtr@record}[3]{}%
276     \let\@do@wrglossary\glsxtr@@do@wrglossary
277     \let\@gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
278     \let\glsxtrundefaction\@glsxtr@err@undefaction
279     \let\glsxtr@warnonexistsordo\@gobble
280     \let\@glsxtr@recordcounter\@glsxtr@noop@recordcounter
281     \def\printunsrtglossaryunit{\print@noop@unsrtglossaryunit}%
282     \undef\glsxtrsetaliasnoindex
283   }%
284   \or

```

Only record (don't index).

```
285 \def\glxtr@setup@record{%
286 \glxtr@autoseeindexfalse
287 \let\@do@seeglossary\glxtr@recordsee
288 \let\glxtr@record\@glxtr@record
289 \let\@do@wrglossary\glxtr@do@record@wrglossary
290 \let\glxtr@saveentrycounter\relax
291 \let\glxtrundefaction\glxtr@warn@undefaction
292 \let\glxtr@warnonexistssordo\glxtr@warn@onexistssordo
293 \glxtr@addloclistfield
294 \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
295 \let\@glxtr@recordcounter\glxtr@op@recordcounter
296 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
```

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

```
297 \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.

```
298 \ifdef\@glxtr@setupsort@none{\@glxtr@setupsort@none}{}%
```

Load glossaries-extra-bib2gls:

```
299 \RequirePackage{glossaries-extra-bib2gls}%
300 }%
301 \or
```

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

```
302 \def\glxtr@setup@record{%
303 \renewcommand*{\@do@seeglossary}{\glxtr@dosee@alsoindex@glossary}%
304 \let\glxtr@record\@glxtr@record
305 \let\@do@wrglossary\glxtr@do@alsoindex@wrglossary
306 \let\glxtr@saveentrycounter\glxtr@indexonly@saveentrycounter
307 \let\glxtrundefaction\glxtr@warn@undefaction
308 \let\glxtr@warnonexistssordo\glxtr@warn@onexistssordo
309 \glxtr@addloclistfield
310 \let\@glxtr@recordcounter\glxtr@op@recordcounter
311 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
312 \undef\glxtrsetaliasnoindex
313 }%
314 \fi
315 }
```

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.

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

```
316 \newcommand*{\@glxtr@docdefval}{0}
```

Need to provide conditional commands that are backward compatible:

if@glxtrdocdef

```
317 \newcommand*{\if@glxtrdocdef}{\ifnum\@glxtr@docdefval>0 }
```

lsxtrdocdeftrue

```
318 \newcommand*{\@glxtrdocdeftrue}{\def\@glxtr@docdefval{1}}
```

lsxtrdocdeffalse

```
319 \newcommand*{\@glxtrdocdeffalse}{\def\@glxtr@docdefval{0}}
```

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

```
320 \define@choicekey{glossaries-extra.sty}{docdef}
321   [ \@glxtr@docdefsetting\@glxtr@docdefval ]%
322   {false,true,restricted}[true]%
323 {%
324   \ifnum\@glxtr@docdefval=2\relax
325     \renewcommand*{\@glsdoifexistsorwarn}{\@glsdoifexists}%
326   \fi
327 }
```

ocdefrestricted

```
328 \newcommand*{\if@glxtrdocdefrestricted}{\ifnum\@glxtr@docdefval=2 }
```

oifexistsorwarn

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

```
329 \newcommand*{\@glsdoifexistsorwarn}{\@glsdoifexistsorwarn}
```

indexcrossrefs

Automatically index cross references at the end of the document

```
330 \define@boolkey{glossaries-extra.sty}[@glxtr]{indexcrossrefs}[true]{%
331   \if@glxtrindexcrossrefs
332   \else
333     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
334   \fi
335 }
```

Switch off since this can increase the build time.

```
336 \@glxtrindexcrossrefsfalse
```

But allow see key to switch it on automatically.

oindexcrossrefs

```
337 \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`.

```

338 \define@boolkey{glossaries-extra.sty}{@glxtr@}{autoseeindex}[true]{%
339 }
340 \@glxtr@autoseeindextrue

```

iesExtraWarning Allow users to suppress warnings.

```

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

```

raWarningNoLine Allow users to suppress warnings.

```

342 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
343 \PackageWarningNoLine{glossaries-extra}{#1}}

344 \@glxtr@declareoption{nowarn}{%
345 \let\GlossariesExtraWarning\@gobble
346 \let\GlossariesExtraWarningNoLine\@gobble
347 \glxtr@doption{nowarn}%
348 }

```

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

```

349 \newcommand*{\@glxtr@defpostpunc}{%

```

postdot Shortcut for `nopostdot=false`

```

350 \@glxtr@declareoption{postdot}{%
351 \glxtr@doption{nopostdot=false}%
352 \renewcommand*{\@glxtr@defpostpunc}{%
353 \renewcommand*{\glspostdescription}{%
354 \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
355 }%
356 }

```

nopostdot Needs to redefine `\@glxtr@defpostpunc`

```

357 \define@choicekey{glossaries-extra.sty}{nopostdot}{true,false}[true]{%
358 \glxtr@doption{nopostdot=#1}%
359 \renewcommand*{\@glxtr@defpostpunc}{%
360 \renewcommand*{\glspostdescription}{%
361 \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
362 }%
363 }

```

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

```

364 \define@key{glossaries-extra.sty}{postpunc}{%
365 \glxtr@doption{nopostdot=false}%
366 \ifstrequal{#1}{dot}%
367 {%
368 \renewcommand*{\@glxtr@defpostpunc}{%

```



```

369     \renewcommand*{\glspostdescription}{.\spacefactor\sfcode'\. }%
370 }%
371 }%
372 {%
373     \ifstrequal{#1}{comma}%
374     {%
375         \renewcommand*{\@glxtr@defpostpunc}{%
376             \renewcommand*{\glspostdescription}{,}%
377         }%
378     }%
379     {%
380         \ifstrequal{#1}{none}%
381         {%
382             \glxtr@dooption{nopostdot=true}%
383             \renewcommand*{\@glxtr@defpostpunc}{%
384                 \renewcommand*{\glspostdescription}{}%
385             }%
386         }%
387         {%
388             \renewcommand*{\@glxtr@defpostpunc}{%
389                 \renewcommand*{\glspostdescription}{#1}%
390             }%
391         }%
392     }%
393 }%
394 }

```

`\glxtrabbrvtype` Glossary type for abbreviations.

```

395 \newcommand*{\glxtrabbrvtype}{\gldefaulttype}

```

`\bbreviationsdef` Set by abbreviations option.

```

396 \newcommand*{\@glxtr@abbreviationsdef}{}

```

`\bbreviationsdef`

```

397 \newcommand*{\@glxtr@doabbreviationsdef}{%
398     \@ifpackageloaded{babel}%
399     {\providecommand{\abbreviationsname}{\acronymname}}%
400     {\providecommand{\abbreviationsname}{Abbreviations}}%
401     \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
402     \renewcommand*{\glxtrabbrvtype}{abbreviations}%
403     \newcommand*{\printabbreviations}[1][1]{%
404         \printglossary[type=\glxtrabbrvtype,##1]%
405     }%
406     \disable@keys{glossaries-extra.sty}{abbreviations}%

```

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

```

407     \ifglsacronym
408     \else
409         \renewcommand*{\acronymtype}{\glxtrabbrvtype}%

```

```

410 \fi
411 }%

```

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

```

412 \@glsxtr@declareoption{abbreviations}{%
413 \let\@glsxtr@abbreviationsdef\@glsxtr@doabbreviationsdef
414 }

```

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

```

415 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
416 \newcommand*{\ab}{\cglsl}%
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 \newcommand*{\AB}{\cGLS}%
433 \newcommand*{\ABP}{\cGLSpl}%
434 \newcommand*{\AS}{\GLSxtrshort}%
435 \newcommand*{\ASP}{\GLSxtrshortpl}%
436 \newcommand*{\AL}{\GLSxtrlong}%
437 \newcommand*{\ALP}{\GLSxtrlongpl}%
438 \newcommand*{\AF}{\GLSxtrfull}%
439 \newcommand*{\AFP}{\GLSxtrfullpl}%

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

```

Disable this command after it's been used.

```

441 \let\GlsXtrDefineAbbreviationShortcuts\relax
442 }

```

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

```

443 \newcommand*{\GlsXtrDefineAcShortcuts}{%
444 \newcommand*{\ac}{\cglsl}%
445 \newcommand*{\acp}{\cglspl}%
446 \newcommand*{\acs}{\glsxtrshort}%

```

```

447 \newcommand*{\acsp}{\glxtrshortpl}%
448 \newcommand*{\acl}{\glxtrlong}%
449 \newcommand*{\aclp}{\glxtrlongpl}%
450 \newcommand*{\acf}{\glxtrfull}%
451 \newcommand*{\acfp}{\glxtrfullpl}%
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 \newcommand*{\AC}{\cGLS}%
461 \newcommand*{\ACP}{\cGLSpl}%
462 \newcommand*{\ACS}{\GLSxtrshort}%
463 \newcommand*{\ACSP}{\GLSxtrshortpl}%
464 \newcommand*{\ACL}{\GLSxtrlong}%
465 \newcommand*{\ACLP}{\GLSxtrlongpl}%
466 \newcommand*{\ACF}{\GLSxtrfull}%
467 \newcommand*{\ACFP}{\GLSxtrfullpl}%

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

  Disable this command after it's been used.
469 \let\GlsXtrDefineAcShortcuts\relax
470 }

```

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

```

471 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
472 \newcommand*{\newentry}{\newglossaryentry}%
473 \ifdef\printsymbols
474 {%
475 \newcommand*{\newsym}{\glxtrnewsymbol}%
476 }{}%
477 \ifdef\printnumbers
478 {%
479 \newcommand*{\newnum}{\glxtrnewnumber}%
480 }{}%
481 \let\GlsXtrDefineOtherShortcuts\relax
482 }

```

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.

```

483 \newcommand*{@glxtr@setupshortcuts}{}

```

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

```
484 \newcommand*{\@glxtr@shortcutsval}{\ifglxtrshortcuts 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).

```
485 \define@choicekey{glossaries-extra.sty}{shortcuts}%
486 [\@glxtr@shortcutsval\@glxtr@shortcutsnr]%
487 {acronyms,acro,abbreviations,abbr,other,all,true,ac,none,false}[true]{%
488   \ifcase\@glxtr@shortcutsnr\relax % acronyms
489     \renewcommand*{\@glxtr@setupshortcuts}{%
490       \glxtrshortcutstrue
491       \DefineAcronymSynonyms
492     }%
493   \or % acro
494     \renewcommand*{\@glxtr@setupshortcuts}{%
495       \glxtrshortcutstrue
496       \DefineAcronymSynonyms
497     }%
498   \or % abbreviations
499     \renewcommand*{\@glxtr@setupshortcuts}{%
500       \GlsXtrDefineAbbreviationShortcuts
501     }%
502   \or % abbr
503     \renewcommand*{\@glxtr@setupshortcuts}{%
504       \GlsXtrDefineAbbreviationShortcuts
505     }%
506   \or % other
507     \renewcommand*{\@glxtr@setupshortcuts}{%
508       \GlsXtrDefineOtherShortcuts
509     }%
510   \or % all
511     \renewcommand*{\@glxtr@setupshortcuts}{%
512       \glxtrshortcutstrue
513       \GlsXtrDefineAcShortcuts
514       \GlsXtrDefineAbbreviationShortcuts
515       \GlsXtrDefineOtherShortcuts
516     }%
517   \or % true
518     \renewcommand*{\@glxtr@setupshortcuts}{%
519       \glxtrshortcutstrue
520       \GlsXtrDefineAcShortcuts
521       \GlsXtrDefineAbbreviationShortcuts
522       \GlsXtrDefineOtherShortcuts
523     }%
```

```

524 \or % ac
525 \renewcommand*{\@glxtr@setupshortcuts}{%
526 \glsacrshortcutstrue
527 \GlsXtrDefineAcShortcuts
528 }%

Leave none and false as last option.
529 \else % none, false
530 \renewcommand*{\@glxtr@setupshortcuts}{}%
531 \fi
532 }

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

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

GlossaryWarning Warning text displayed in document if the external glossary file given by the argument is missing.
536 \newcommand{\glxtrNoGlossaryWarning}[1]{%
537 \@glxtr@defaultnoglossarywarning{#1}%
538 }

omissingglstext If true, suppress the text produced if the external glossary file is missing.
539 \define@choicekey{glossaries-extra.sty}{nomissingglstext}
540 [\@glxtr@nomissingglstextval\@glxtr@nomissingglstextnr]%
541 {true,false}[true]{%
542 \ifcase\@glxtr@nomissingglstextnr\relax % true
543 \renewcommand{\glxtrNoGlossaryWarning}[1]{\null}%
544 \else % false
545 \renewcommand{\glxtrNoGlossaryWarning}[1]{%
546 \@glxtr@defaultnoglossarywarning{#1}%
547 }%
548 \fi
549 }

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

xtr@redefstyles
550 \newcommand*{\@glxtr@redefstyles}{}

stylemods
551 \define@key{glossaries-extra.sty}{stylemods}[default]{%
552 \ifstrequal{#1}{default}%
553 {%
554 \renewcommand*{\@glxtr@redefstyles}{%
555 \RequirePackage{glossaries-extra-stylemods}}%

```

```

556 }%
557 {%
558   \ifstrequal{#1}{all}%
559   {%
560     \renewcommand*{\@glxtr@redefstyles}{%
561       \PassOptionsToPackage{all}{glossaries-extra-stylemods}%
562       \RequirePackage{glossaries-extra-stylemods}%
563     }%
564   }%
565   {%
566     \renewcommand*{\@glxtr@redefstyles}{}%
567     \@for\@glxtr@tmp:=#1\do{%
568       \IfFileExists{glossary-\@glxtr@tmp.sty}%
569       {%
570         \eappto\@glxtr@redefstyles{%
571           \noexpand\RequirePackage{glossary-\@glxtr@tmp}}%
572         }%
573       {%
574         \PackageError{glossaries-extra}%
575           {Glossaries style package ‘glossary-\@glxtr@tmp.sty’
576             doesn’t exist (did you mean to use the ‘style’ key?)}%
577           {The list of values (#1) in the ‘stylemods’ key should
578             match the glossary-xxx.sty files provided with
579             glossaries.sty}%
580       }%
581     }%
582     \appto\@glxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
583   }
584 }%
585 }

```

glxtr@do@style

```

586 \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.

```

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

```

Defer actual style change:

```

588 \renewcommand*{\@glxtr@do@style}{%

```

Set this as the default style:

```

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

```

Set this style:

```

590 \setglossarystyle{#1}%

```

```

591 }%

```

```

592 }

```

`c@wrglossaryctr` Increments the associated counter if enabled. Does nothing by default. The optional argument is the entry label in case it's required, but the `wrglossary` counter is globally used by all entries.

```
593 \newcommand*{\glstr@inc@wrglossaryctr}[1]{}
```

`ocationHyperlink`

```
\glstrinternallocationhyperlink{<counter>}{<prefix>}{<location>}
```

The first two arguments are always control sequences.

```
594 \newcommand*{\GlsXtrInternalLocationHyperlink}[3]{%
595   \glstrhyperlink{#1#2#3}{#3}%
596 }
```

`cationhyperlink`

```
597 \newcommand*{@glstr@wrglossary@locationhyperlink}[3]{%
598   \pageref{wrglossary.#3}%
599 }
```

`indexcounter`

Define the `wrglossary` counter that's incremented every time an entry is indexed, except for cross-references. This is designed for use with `bib2gls v1.4+`. It can work with the other indexing methods but it will interfere with the number list collation. This option automatically implements `counter=wrglossary`.

Since glossaries automatically loads `amsmath`, there may be a problem if the indexing occurs in the equation environment, because only one `\label` is allowed in each instance of that environment. It's best to change the counter when in maths mode.

```
600 \@glstr@declareoption{indexcounter}{%
601   \glstr@doooption{counter=wrglossary}%
602   \ifundef\c@wrglossary
603   {%
604     \newcounter{wrglossary}%
605     \renewcommand{\thewrglossary}{\arabic{wrglossary}}%
606   }%
607   {}%
608   \renewcommand*{\glstr@inc@wrglossaryctr}[1]{%
```

Only increment if the current counter is `wrglossary`.

```
609   \ifdefstring\@gls@counter{wrglossary}%
610   {%
611     \refstepcounter{wrglossary}%
612     \label{wrglossary.\thewrglossary}%
613   }%
614   {}%
615 }%
616 \renewcommand*{\GlsXtrInternalLocationHyperlink}[3]{%
617   \ifdefstring\glsentrycounter{wrglossary}%
618   {%
```

```

619     \@glsxtr@wrglossary@locationhyperlink{##1}{##2}{##3}%
620   }%
621   {\glsxtrhyperlink{##1##2##3}{##3}}%
622 }%
623 }

```

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

```

624 \newcommand*{\@glsxtrwrglossmark}{}

```

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

```

625 \newcommand*{\@@glsxtrwrglossmark}{}
626 \AtBeginDocument{\renewcommand*{\@glsxtrwrglossmark}{\@glsxtrwrglossmark}}

```

sxtrwrglossmark Does nothing by default.

```

627 \newcommand*{\glsxtrwrglossmark}{\ensuremath{\cdot}}

```

debug Provide extra debug options.

```

628 \define@choicekey{glossaries-extra.sty}{debug}
629 [ \@glsxtr@debugval \@glsxtr@debugnr ]%
630 {true,false,showtargets,showwrgloss,all}[true]{%
631   \ifcase\@glsxtr@debugnr\relax % true
632     \glsxtr@doption{debug=true}%
633     \renewcommand*{\@glsxtrwrglossmark}{}%
634   \or % false
635     \glsxtr@doption{debug=false}%
636     \renewcommand*{\@glsxtrwrglossmark}{}%
637   \or % showtargets
638     \glsxtr@doption{debug=showtargets}%
639   \or % showwrgloss
640     \glsxtr@doption{debug=true}%
641     \renewcommand*{\@glsxtrwrglossmark}{\glsxtrwrglossmark}%
642   \or % all
643     \glsxtr@doption{debug=showtargets}%
644     \renewcommand*{\@glsxtrwrglossmark}{\glsxtrwrglossmark}%
645   \fi
646 }

```

Pass all other options to glossaries.

```

647 \DeclareOptionX*{%
648   \expandafter\glsxtr@doption\expandafter{\CurrentOption}}

```

Process options.

```

649 \ProcessOptionsX

```

Load glossaries if not already loaded.

```

650 \RequirePackage{glossaries}

```

Load the glossaries-accsupp package if required.

```

651 \@glsxtr@doaccsupp

```


Redefine \glspostdescription if required.

```
652 \@glstr@defpostpunc
```

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

```
653 \def\glsshowtarget#1{%
654   \glstrtitleorpdforheading
655   {%
656     \ifmmode
657       \texttt{\small [#1]}%
658     \else
659       \ifinner
660         \texttt{\small [#1]}%
661       \else
662         \marginpar{\texttt{\small #1}}%
663       \fi
664     \fi
665   }%
666   {[#1]}%
667   {\texttt{\small [#1]}}%
668 }
```

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

```
669 \let\@glstr@org@doseeglossary\@do@seeglossary
```

r@doseeglossary This doesn't increment the associated counter.

```
670 \newcommand*{\@glstr@doseeglossary}[2]{%
671   \glsdofexists{#1}%
672   {%
673     \@glstrwrglossmark
674     \@glstr@org@doseeglossary{#1}{#2}%
675   }%
676 }
```

oindex@glossary

```
677 \newcommand*{\@glstr@dosee@alsoindex@glossary}[2]{%
678   \@glstr@recordsee{#1}{#2}%
679   \@glstr@doseeglossary{#1}{#2}%
680 }
```

@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.)

```
681 \let\@glstr@org@gloautosee\@glo@autosee
```

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

```
682 \if@glstr@autoseeindex
683 \else
```

```

684 \ifdef\@glxtr@org@gloautosee
685 {}%
686 {\PackageError{glossaries-extra}{‘autoseeindex=false’ package
687 option requires at least v4.30 of glossaries.sty}%
688 {You need to update the glossaries.sty package}%
689 }
690 \fi

```

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

```

691 \ifdef\@glo@autosee
692 {%
693 \renewcommand*{\@glo@autosee}{%
694 \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
695 }%
696 {}

```

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.

```

697 \renewcommand*{\gls@checkseeallowed}{%
698 \if@glxtr@autoseeindex\@gls@see@noindex\fi
699 }

```

Define abbreviations glossaries if required.

```

700 \@glxtr@abbreviationsdef
701 \let\@glxtr@abbreviationsdef\relax

```

Setup shortcuts if required.

```

702 \@glxtr@setupshortcuts

```

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

```

703 \@glxtr@redef@for@gl@sentries

```

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

```

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

```

Now define the user command:

```

705 \newcommand*{\glossariesextrasetup}[1]{%
706 \let\glxtr@setup@record\relax
707 \let\@glxtr@setupshortcuts\relax
708 \let\@glxtr@redef@for@gl@sentries\relax
709 \setkeys{glossaries-extra.sty}{#1}%
710 \@glxtr@abbreviationsdef
711 \let\@glxtr@abbreviationsdef\relax
712 \@glxtr@setupshortcuts
713 \glxtr@setup@record
714 \@glxtr@redef@for@gl@sentries
715 }

```

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

```
716 \let\glxstr@org@@do@wrglossary\@@do@wrglossary
```

`@@do@wrglossary` The new version adds code that can show a marker for debugging and increments the associated counter if enabled.

```
717 \newcommand*{\glxstr@@do@wrglossary}[1]{%
718   \@@glxstrwrglossmark
719   \glxstr@inc@wrglossaryctr{#1}%
720   \glxstr@org@@do@wrglossary{#1}%
721 }
```

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

```
722 \let\glxstr@saveentrycounter\@gls@saveentrycounter
```

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

```
723 \let\@gls@saveentrycounter\glxstr@indexonly@saveentrycounter
```

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

`sxtrdialecthook`

```
724 \newcommand*{\@glxstrdialecthook}{}
```

Set up record option if required.

```
725 \glxstr@setup@record
```

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

```
726 \AtBeginDocument{%
727   \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
728   \def\@glxstrundeftag{\glxstrundeftag}%
729 }
```

1.2 Extra Utilities

`rifemptyglossary` `\glxstrifemptyglossary{<type>}{<true>}{<false>}`

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

```
730 \newcommand{\glxstrifemptyglossary}[3]{%
731   \ifcsdef{glolist@#1}%
```

```

732  {%
733    \ifcsstring{glolist@#1}{,}{#2}{#3}%
734  }%
735  {%
736    \glstrundefaction{Glossary type ‘#1’ doesn’t exist}{}%
737    #2%
738  }%
739 }

```

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

```

740 \newcommand*{\glxtrifkeydefined}[3]{%
741   \key@ifundefined{glossentry}{#1}{#3}{#2}%
742 }

```

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

```

743 \newcommand*{\glxtrprovidestoragekey}{%
744   \@ifstar\sglsxtr@provide@storagekey\@glxtr@provide@storagekey
745 }

```

vide@storagekey Unstarred version.

```

746 \newcommand*{\@glxtr@provide@storagekey}[3]{%
747   \key@ifundefined{glossentry}{#1}%
748   {%
749     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
750     \appto\@gls@keymap{,}{#1}{#1}}%
751     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
752     \appto\@newglossaryentryposthook{%
753       \letcs{\@glo@tmp}{@glo@#1}%
754       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
755     }%

```

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

```

756   \ifblank{#3}
757   {}%
758   {%
759     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
760   }%
761 }%
762 {%

```

Provide the no-link command if not already defined.

```

763   \ifblank{#3}
764   {}%
765   {%
766     \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
767   }%
768 }%
769 }

```

vide@storagekey Starred version.

```
770 \newcommand*{\s@glxtr@provide@storagekey}[1]{%
771   \key@ifundefined{glossentry}{#1}%
772   {%
773     \expandafter\newcommand\expandafter*\expandafter
774     {\csname gls@assign@#1@field\endcsname}[2]{%
775       \@@gls@expand@field{##1}{#1}{##2}%
776     }%
777   }%
778   {}%
779   \@glxtr@provide@addstoragekey{#1}%
780 }
```

The name of a text-block control sequence can be stored in a field (given by `\GlsXtrFmtField`). This command can then be used with `\glxtrfmt[<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`

```
781 \newcommand{\GlsXtrFmtField}{useri}
```

`\glxtrfmt`

```
782 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}
```

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

```
783 \newrobustcmd*{\glxtrfmt}{\@ifstar\s@glxtrfmt\@glxtrfmt}
```

`\@glxtrfmt` Unstarred form.

```
784 \newcommand*{\@glxtrfmt}[3][\@glxtrfmt{#1}{#2}{#3}]{}
```

`\s@glxtrfmt` Starred form.

```
785 \newcommand*{\s@glxtrfmt}[3][\@glxtrfmt{#1}{#2}{#3}]{%
786   \new@ifnextchar[\s@@glxtrfmt{#1}{#2}{#3}]{%
787     {\@glxtrfmt{#1}{#2}{#3}}}%
788 }
```

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

```
789 \def\s@@glxtrfmt#1#2#3[#4]{\@glxtrfmt{#1}{#2}{#3}{#4}}
```

`\@@glxtrfmt` Actual inner working.

```
790 \newcommand*{\@@glxtrfmt}[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).

```
791 \begingroup
792   \def\glslabel{#2}%
793   \glsdoifexistsordo{#2}%
```

```

794   {%
795     \ifglshasfield{\GlsXtrFmtField}{#2}%
796     {%
797       \let\do@gl@link@checkfirsthyper\relax
798       \expandafter\@gl@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
799       {\glxtrfmtdisplay{\glscurrentfieldvalue}{#3}{#4}}%
800     }%
801     {\glxtrfmtdisplay{@firstofone}{#3}{#4}}%
802   }%
803   {%

```

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

```

804     \begingroup
805       \@gl@setdefault@gl@link@opts
806       \setkeys{gl@link}{\GlsXtrFmtDefaultOptions,#1}%
807       \ifKV@gl@link@noindex\else\gl@sadd{#2}\fi
808     \endgroup
809     \glxtrfmtdisplay{@firstofone}{#3}{#4}%
810   }%
811 \endgroup
812 }

```

glxtrfmtdisplay The command used internally by \glxtrfmt 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).

```

813 \newcommand{\glxtrfmtdisplay}[3]{\csuse{#1}{#2}#3}

```

glxtrentryfmt No link or indexing.

```

814 \ifdef\teorpdfstring
815 {
816   \newcommand*{\glxtrentryfmt}[2]{%
817     \teorpdfstring{\@glxtrentryfmt{#1}{#2}}{#2}%
818   }
819 }
820 {
821   \newcommand*{\glxtrentryfmt}{\@glxtrentryfmt}
822 }

```

@glxtrentryfmt

```

823 \newrobustcmd*{\@glxtrentryfmt}[2]{%
824   \gl@sdoifexistsordo{#1}%
825   {%
826     \ifglshasfield{\GlsXtrFmtField}{#1}%
827     {%
828       \csuse{\glscurrentfieldvalue}{#2}%
829     }%
830     {#2}%
831   }%

```

```
832 {#2}%
833 }
```

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

```
834 \newcommand*{\glstrfieldlistadd}[3]{%
835   \listcsadd{glo@\glstetoklabel{#1}@#2}{#3}%
836 }
```

`glstrfieldlistgadd` Similarly but uses `\listcsgadd`.

```
837 \newcommand*{\glstrfieldlistgadd}[3]{%
838   \listcsgadd{glo@\glstetoklabel{#1}@#2}{#3}%
839 }
```

`glstrfieldlistseadd` Similarly but uses `\listcseadd`.

```
840 \newcommand*{\glstrfieldlistseadd}[3]{%
841   \listcseadd{glo@\glstetoklabel{#1}@#2}{#3}%
842 }
```

`glstrfieldlistxadd` Similarly but uses `\listcsxadd`.

```
843 \newcommand*{\glstrfieldlistxadd}[3]{%
844   \listcsxadd{glo@\glstetoklabel{#1}@#2}{#3}%
845 }
```

Now provide commands to iterate over these lists.

`glstrfielddolistloop`

```
846 \newcommand*{\glstrfielddolistloop}[2]{%
847   \dolistcsloop{glo@\glstetoklabel{#1}@#2}%
848 }
```

`glstrfieldforlistloop`

```
849 \newcommand*{\glstrfieldforlistloop}[3]{%
850   \forlistcsloop{#3}{glo@\glstetoklabel{#1}@#2}%
851 }
```

List element tests:

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

```
852 \newcommand*{\glstrfieldifinlist}[5]{%
853   \ifinlistcs{#3}{glo@\glstetoklabel{#1}@#2}{#4}{#5}%
854 }
```

`glstrfieldxifinlist` Expands item.

```
855 \newcommand*{\glstrfieldxifinlist}[5]{%
856   \xifinlistcs{#3}{glo@\glstetoklabel{#1}@#2}{#4}{#5}%
857 }
```

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

```
858 \newcommand*{\glxtrforcsvfield}[3]{%
859   \@glxtrifhasfield{#2}{#1}%
860   {%
861     \let\glxxtrendfor\@endfortrue
862     \@for\@glxtr@label:=\glscurrentfieldvalue\do
863     {\expandafter#3\expandafter{\@glxtr@label}}}%
864   }%
865 }
```

`\glxtrifhasfield` 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.

```
866 \newrobustcmd{\glxtrifhasfield}{%
867   \@ifstar{\s@glxtrifhasfield}{\@glxtrifhasfield}%
868 }
```

`\glxtrifhasfield` Unstarred version adds grouping.

```
869 \newcommand{\@glxtrifhasfield}[4]{%
870   {\s@glxtrifhasfield{#1}{#2}{#3}{#4}}%
871 }
```

`\glxtrifhasfield` Starred version omits grouping.

```
872 \newcommand{\s@glxtrifhasfield}[4]{%
873   \letcs{\glscurrentfieldvalue}{glo@\glsdetoklabel{#2}@#1}%
874   \ifundef\glscurrentfieldvalue
875     {#4}%
876     {%
877       \ifdefempty\glscurrentfieldvalue{#4}{#3}%
878     }%
879 }
```

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

Just uses `\ifcsundef`.

```
880 \newcommand{\GlsXtrIfFieldUndef}[2]{%
881   \ifcsundef{glo@\glsdetoklabel{#2}@#1}%
882 }
```

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

```
883 \newcommand*{\glxtrusefield}[2]{%
884   \@glsc@entry@field{#1}{#2}%
885 }
```


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

```

886 \newcommand*{\Glsxtrusefield}[2]{%
887   \@gls@entry@field{#1}{#2}%
888 }

```

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

```

889 \newcommand*{\glsxtrdeffield}[2]{\csdef{glo@\glsdetoklabel{#1}@#2}}

```

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

```

890 \newcommand*{\glsxtredefield}[2]{\protected@csedef{glo@\glsdetoklabel{#1}@#2}}

```

`etfieldifexists`

```

891 \newcommand*{\glsxtrsetfieldifexists}[3]{\glsdoifexists{#1}{#3}}

```

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

```

892 \newrobustcmd*{\GlsXtrSetField}[3]{%
893   \glsxtrsetfieldifexists{#1}{#2}%
894   {\csdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
895 }

```

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

```

896 \newrobustcmd*{\GlsXtrLetField}[3]{%
897   \glsxtrsetfieldifexists{#1}{#2}%
898   {\cslet{glo@\glsdetoklabel{#1}@#2}{#3}}%
899 }

```

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

```

900 \newrobustcmd*{\csGlsXtrLetField}[3]{%
901   \glsxtrsetfieldifexists{#1}{#2}%
902   {\csletcs{glo@\glsdetoklabel{#1}@#2}{#3}}%
903 }

```

`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.

```

904 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
905   \glsxtrsetfieldifexists{#1}{#2}%
906   {\csletcs{glo@\glsdetoklabel{#1}@#2}{glo@\glsdetoklabel{#3}@#4}}%
907 }

```

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

```

908 \newrobustcmd*{\gGlsXtrSetField}[3]{%
909   \glsxtrsetfieldifexists{#1}{#2}%
910   {\csgdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
911 }

```

xGlsXtrSetField

```
912 \newrobustcmd*{\xGlsXtrSetField}[3]{%
913   \glstrsetfieldifexists{#1}{#2}%
914   {\protected@csxdef{glo\xGlsdetoklabel{#1}@#2}{#3}}%
915 }
```

eGlsXtrSetField

```
916 \newrobustcmd*{\eGlsXtrSetField}[3]{%
917   \glstrsetfieldifexists{#1}{#2}%
918   {\protected@csedef{glo\xGlsdetoklabel{#1}@#2}{#3}}%
919 }
```

XtrIfFieldEqStr

```
920 \newrobustcmd*{\GlsXtrIfFieldEqStr}[5]{%
921   \glstrifhasfield{#1}{#2}%
922   {%
923     \ifdefstring{\glscurrentfieldvalue}{#3}{#4}{#5}%
924   }%
925   {#5}%
926 }
```

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

```
927 \ifglssentrycounter
928   \newcommand*{\glstrpageref}[1]{\pageref{glssentry-\Glsdetoklabel{#1}}}
929 \else
930   \ifglssubentrycounter
931     \newcommand*{\glstrpageref}[1]{\pageref{glssentry-\Glsdetoklabel{#1}}}
932   \else
933     \newcommand*{\glstrpageref}[1]{\glss{#1}}
934   \fi
935 \fi
```

lossarypreamble

```
936 \newcommand{\apptoglossarypreamble}[2][\glssdefaulttype]{%
937   \ifcsdef{glolist@#1}%
938   {%
939     \ifcsundef{@glossarypreamble@#1}%
940     {\csdef{@glossarypreamble@#1}{}}%
941   }%
942   \csappto{@glossarypreamble@#1}{#2}%
943   {%
944     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
945   }%
946 }%
947 }
```

lossarypreamble

```
948 \newcommand{\preglossarypreamble}[2][\glssdefaulttype]{%
```

```

949 \ifcsdef{glolist@#1}%
950 {%
951   \ifcsundef{@glossarypreamble@#1}%
952   {\csdef{@glossarypreamble@#1}{}}%
953   {}%
954   \cspretto{@glossarypreamble@#1}{#2}%
955   }%
956   {%
957     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
958   }%
959 }

```

1.3 Modifications to Commands Provided by glossaries

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

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

`\newglossaryentry`

```

960 \renewcommand*{\longnewglossaryentry}{%
961   \@ifstar{\glstr@s@longnewglossaryentry}\glstr@longnewglossaryentry
962 }

```

`\newglossaryentry` Starred version.

```

963 \newcommand{\@glstr@s@longnewglossaryentry}[3]{%
964   \glsdoifnoexists{#1}%
965   {%
966     \bgroup
967     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
968     \long\def\@newglossaryentryprehook{%
969       \long\def\@glo@desc{#3}%
970       \@org@newglossaryentryprehook
971     }%
972     \renewcommand*{\gls@assign@desc}[1]{%
973       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
974       \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
975     }
976     \gls@defglossaryentry{#1}{#2}%
977   \egroup
978   }%
979 }

```

`\newglossaryentry` Unstarred version.

```

980 \newcommand{\@glstr@longnewglossaryentry}[3]{%
981   \glsdoifnoexists{#1}%

```

```

982  {%
983    \bgroup
984      \let\@org@newglossaryentryprehook\@newglossaryentryprehook
985      \long\def\@newglossaryentryprehook{%
986        \long\def\@glo@desc{#3\glstrpostlongdescription}%
987        \@org@newglossaryentryprehook
988      }%
989      \renewcommand*{\gl@assign@desc}[1]{%
990        \global\cslet{glo@\gl@detoklabel{#1}@desc}{\@glo@desc}%

```

The following is different from the base glossaries.sty:

```

991      \global\cslet{glo@\gl@detoklabel{#1}@descplural}{\@glo@descplural}%
992    }
993    \gl@defglossaryentry{#1}{#2}%
994  \egroup
995 }%
996 }

```

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

```

997 \newcommand*{\glstrpostlongdescription}{\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.

```

998 \renewcommand{\newignoredglossary}{%
999   \@ifstar\glstr@s@newignoredglossary\glstr@org@newignoredglossary
1000 }

```

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

```

1001 \newcommand*{\glstr@org@newignoredglossary}[1]{%
1002   \ifcsdef{glolist@#1}
1003   {%
1004     \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1005   }%
1006   {%
1007     \ifdefempty\@ignored@glossaries
1008     {%
1009       \edef\@ignored@glossaries{#1}%
1010     }%
1011     {%
1012       \eappto\@ignored@glossaries{,#1}%
1013     }%
1014     \csgdef{glolist@#1}{,}%
1015     \ifcsundef{gl@#1@entryfmt}%
1016     {%
1017       \defgl@entryfmt[#1]{\gl@entryfmt}%
1018     }%
1019     {}%

```

```

1020 \ifdefempty\@gls@nohyperlist
1021 {%
1022   \renewcommand*{\@gls@nohyperlist}{#1}%
1023   }%
1024   {%
1025     \eappto\@gls@nohyperlist{, #1}%
1026     }%
1027   }%
1028 }

```

ignoredglossary Starred form.

```

1029 \newcommand*{\glstr@s@newignoredglossary}[1]{%
1030   \ifcsdef{glolist@#1}
1031   {%
1032     \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1033   }%
1034   {%
1035     \ifdefempty\@ignored@glossaries
1036     {%
1037       \edef\@ignored@glossaries{#1}%
1038     }%
1039     {%
1040       \eappto\@ignored@glossaries{, #1}%
1041     }%
1042     \csgdef{glolist@#1}{,}%
1043     \ifcsundef{gls@#1@entryfmt}%
1044     {%
1045       \defglsentryfmt[#1]{\glsentryfmt}%
1046     }%
1047     {}%
1048   }%
1049 }

```

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

```

1050 \glsifusetranslator
1051 {%
1052   \renewcommand*{\glstoctitle}[1]{%
1053     \ifcsdef{gls@tr@set@#1@toctitle}%
1054     {%
1055       \csuse{gls@tr@set@#1@toctitle}%
1056     }%
1057     {%
1058       \ifcsdef{@glotype@#1@title}%
1059       {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1060       {\def\glossarytoctitle{\glossarytitle}}%
1061     }%
1062   }%
1063 }

```

```

1064 {
1065   \renewcommand*{\glstocctitle}[1]{%
1066     \ifcsdef{@glotype@#1@title}%
1067       {\def\glossarytocctitle{\csname @glotype@#1@title\endcsname}}%
1068       {\def\glossarytocctitle{\glossarytitle}}%
1069   }
1070 }

```

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

```

1071 \newcommand{\provideignoredglossary}{%
1072   \@ifstar\glxtr@s@provideignoredglossary\glxtr@provideignoredglossary
1073 }

```

ignoredglossary Unstarred version.

```

1074 \newcommand*{\glxtr@provideignoredglossary}[1]{%
1075   \ifcsdef{glolist@#1}
1076   {}%
1077   {%
1078     \ifdefempty\@ignored@glossaries
1079     {%
1080       \edef\@ignored@glossaries{#1}%
1081     }%
1082     {%
1083       \eappto\@ignored@glossaries{,#1}%
1084     }%
1085     \csgdef{glolist@#1}{,}%
1086     \ifcsundef{gls@#1@entryfmt}%
1087     {%
1088       \defglsentryfmt[#1]{\glsentryfmt}%
1089     }%
1090     {}%
1091     \ifdefempty\@gls@nohyperlist
1092     {%
1093       \renewcommand*{\@gls@nohyperlist}{#1}%
1094     }%
1095     {%
1096       \eappto\@gls@nohyperlist{,#1}%
1097     }%
1098   }%
1099 }

```

ignoredglossary Starred form.

```

1100 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
1101   \ifcsdef{glolist@#1}
1102   {}%
1103   {%
1104     \ifdefempty\@ignored@glossaries
1105     {%
1106       \edef\@ignored@glossaries{#1}%

```

```

1107 }%
1108 {%
1109     \eappto\@ignored@glossaries{,#1}%
1110 }%
1111 \csgdef{glolist@#1}{,}%
1112 \ifcsundef{gls@#1@entryfmt}%
1113 {%
1114     \defglsentryfmt[#1]{\glsentryfmt}%
1115 }%
1116 {}%
1117 }%
1118 }

```

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

```

1119 \newcommand*{\glstrcopytoglossary}[2]{%
1120     \glsdoifexists{#1}%
1121     {%
1122         \ifcsdef{glolist@#2}
1123         {%
1124             \cseappto{glolist@#2}{#1,}%
1125         }%
1126         {%
1127             \glstrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
1128         }%
1129     }%
1130 }

```

1.3.1 Existence Checks

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

```

1131 \renewcommand{\glsdoifexists}[2]{%
1132     \ifglsentryexists{#1}{#2}%
1133     {%

```

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

```

1134     \edef\glslabel{\glsdetoklabel{#1}}%
1135     \glstrundefaction{Glossary entry ‘\glslabel’
1136     has not been defined}{You need to define a glossary entry before
1137     you can reference it.}%
1138     }%
1139 }

```

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

```

1140 \renewcommand{\glsdoifnoexists}[2]{%
1141     \ifglsentryexists{#1}{%
1142         \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
1143         has already been defined}{}}{#2}%

```

```
1144 }
```

`\doifexistsordo` 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.

```
1145 \ifdef\glsdoifexistsordo
1146 {%
1147   \renewcommand{\glsdoifexistsordo}[3]{%
1148     \ifglstryexists{#1}{#2}%
1149     {%
1150       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
1151       has not been defined}{You need to define a glossary entry
1152       before you can use it.}%
1153       #3%
1154     }%
1155   }%
1156 }
1157 {%
1158   \glstr@warnonexistsordo\glsdoifexistsordo
1159   \newcommand{\glsdoifexistsordo}[3]{%
1160     \ifglstryexists{#1}{#2}%
1161     {%
1162       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
1163       has not been defined}{You need to define a glossary entry
1164       before you can use it.}%
1165       #3%
1166     }%
1167   }%
1168 }
```

`\arynoexistsordo` Similarly for `\doifglossarynoexistsordo`.

```
1169 \ifdef\doifglossarynoexistsordo
1170 {%
1171   \renewcommand{\doifglossarynoexistsordo}[3]{%
1172     \ifglossaryexists{#1}%
1173     {%
1174       \glstrundefaction{Glossary type '#1' already exists}{}%
1175       #3%
1176     }%
1177     {#2}%
1178   }%
1179 }
1180 {%
1181   \glstr@warnonexistsordo\doifglossarynoexistsordo
1182   \newcommand{\doifglossarynoexistsordo}[3]{%
1183     \ifglossaryexists{#1}%
1184     {%
1185       \glstrundefaction{Glossary type '#1' already exists}{}%
1186       #3%
1187     }%
1188   }%
1189 }
```



```

1188     {#2}%
1189 }%
1190 }
1191

```

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.

```

1192 \appto\@newglossaryentryposthook{%
1193   \ifdefvoid\@glo@see
1194     {\csxdef{glo@\@glo@label @see}{}}%
1195     {%
1196       \csxdef{glo@\@glo@label @see}{\@glo@see}%
1197       \ifglxtr@autoseeindex
1198         \@glxtr@autoindexcrossrefs
1199       \fi
1200     }%
1201 }
1202 \appto\@gls@keymap{,{see}{see}}

```

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

```

1203 \newcommand*{\glxtrusesee}[1]{%
1204   \glsdoifexists{#1}%
1205   {%
1206     \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@see}%
1207     \ifdefempty\@glo@see
1208       {}%
1209     {%
1210       \expandafter\glxtr@usesee\@glo@see\end@glxtr@usesee
1211     }%
1212   }%
1213 }

```

\glxtr@usesee

```

1214 \newcommand*{\glxtr@usesee}[1][\seename]{%
1215   \@glxtr@usesee{#1}%
1216 }

```

\@glxtr@usesee

```

1217 \def\@glxtr@usesee[#1]#2\end@glxtr@usesee{%
1218   \glxtruseseeformat{#1}{#2}%
1219 }

```

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.

```

1220 \newcommand*{\glxtruseseeformat}[2]{%

```

```

1221 \glsseeformat[#1]{#2}{}%
1222 }

```

`\glsseeitemformat` glossaries originally defined `\glsseeitemformat` to use `\glsentryname` but in v3.0 this was switched to use `\glsentrytext` due to problems occurring with the name field being sanitized. Since this is no longer a problem, glossaries-extra restores the original definition as it makes more sense to use the name in the cross-reference list. This still uses `\glsaccesstext` for abbreviations.

```

1223 \renewcommand*{\glsseeitemformat}[1]{%
1224 \ifglshashshort{\glslabel}{\glsaccesstext{#1}}{\glsaccessname{#1}}%
1225 }

```

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

```

1226 \newcommand*{\glsxtruseealso}[1]{%
1227 \glsdoifexists{#1}%
1228 {%
1229 \letcs{\@glo@see}{glo\glsdetoklabel{#1}@seealso}%
1230 \ifdefempty\@glo@see
1231 {}%
1232 {%
1233 \expandafter\glsxtruseealsoformat\expandafter{\@glo@see}%
1234 }%
1235 }%
1236 }

```

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

```

1237 \newcommand*{\glsxtruseealsoformat}[1]{%
1238 \glsseeformat[\seealso]{#1}{}%
1239 }

```

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

```

1240 \newrobustcmd{\glsxtrseelist}[1]{%
1241 \edef\@glo@tmp{\noexpand\glsseelist{#1}}\@glo@tmp
1242 }

```

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

```

1243 \providecommand{\seealso}{}

```

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

```

1244 \ifdef\@xdycrossrefhook
1245 {

```

Add the cross-reference class definition to the hook.

```

1246 \appto\@xdycrossrefhook{%
1247   \write\glswrite{(define-crossref-class \string"seealso\string"
1248     :unverified )}%
1249   \write\glswrite{(markup-crossref-list
1250     :class \string"seealso\string"^^J\space\space\space
1251     :open \string"\string\glstruseealsoformat\glsopenbrace\string"
1252     :close \string"\glsclosebrace\string")}%
1253 }

```

Append to class list.

```

1254 \appto\@xdylocationclassorder{\space\string"seealso\string"}

```

This essentially works like `\do@seeglossary` but uses the `seealso` class. This doesn't increment the associated counter.

```

1255 \newrobustcmd*{\glstrindexseealso}[2]{%
1256   \ifx\@glstr@record@setting\@glstr@record@setting@alsoindex
1257     \@glstr@recordsee{#1}{#2}%
1258   \fi
1259   \glsdoifexists{#1}%
1260   {%
1261     \@@glstrwrglossmark
1262     \def\@gls@xref{#2}%
1263     \@onelevel@sanitize\@gls@xref
1264     \@gls@checkmkidxchars\@gls@xref
1265     \gls@glossary{\csname glo@#1@type\endcsname}{%
1266       (indexentry
1267         :tkey (\csname glo@#1@index\endcsname)
1268         :xref (\string"\@gls@xref\string")
1269         :attr \string"seealso\string"
1270       )
1271     }%
1272   }%
1273 }
1274 }
1275 {

```

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

```

1276 \newrobustcmd*{\glstrindexseealso}{\glssee[\seealsoname]}
1277 }

```

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`.

```

1278 \ifdef\gls@set@xr@key
1279 {

```

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

```

1280 \define@key{glossentry}{alias}{%
1281   \gls@set@xr@key{alias}{\@glo@alias}{#1}%
1282 }
1283 \define@key{glossentry}{seealso}{%
1284   \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
1285 }

```

Add to the key mappings.

```

1286 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}

```

Set the default value.

```

1287 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%

```

Assign the field values.

```

1288 \appto\@newglossaryentryposthook{%
1289   \ifdefvoid\@glo@seealso
1290     {\csxdef{glo@\@glo@label @seealso}{}}%
1291     {%
1292       \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1293       \ifglxtr@autoseeindex
1294         \@glxtr@autoindexcrossrefs
1295       \fi
1296     }%

```

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

```

1297   \ifdefvoid\@glo@alias
1298     {\csxdef{glo@\@glo@label @alias}{}}%
1299     {%
1300       \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1301     }%
1302 }

```

Provide user-level commands to access the values.

\glxtralias

```

1303 \newcommand*\glxtralias[1]{\@gls@entry@field{#1}{alias}}

```

trseealsolabels

```

1304 \newcommand*\glxtrseealsolabels[1]{\@gls@entry@field{#1}{seealso}}

```

Add to the \@glo@autosee hook.

```

1305 \appto\@glo@autoseehook{%
1306   \ifdefvoid\@glo@alias
1307     {%
1308       \ifdefvoid\@glo@seealso
1309         {}%
1310       {%
1311         \edef\@do@glsee{\noexpand\glxtrindexseealso

```

```

1312      {\@glo@label}{\@glo@seealso}}%
1313      \@do@glssee
1314    }%
1315  }%
1316  {%

```

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

```

1317      \ifdefvoid\@glo@see
1318      {%
1319        \edef\@do@glssee{\noexpand\glssee{\@glo@label}{\@glo@alias}}%
1320        \@do@glssee
1321      }%
1322    }%
1323  }%
1324 }%
1325 }
1326 {

```

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

\glsxtralias

```

1327 \glsaddstoragekey*{alias}{\glsxtralias}

```

trseealsolabels

```

1328 \glsaddstoragekey*{seealso}{\glsxtrseealsolabels}

```

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

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

```

1329 \appto\@newglossaryentryposthook{%
1330   \ifcsvoid{glo@\@glo@label @alias}%
1331   {%
1332     \ifcsvoid{glo@\@glo@label @seealso}%
1333     }%
1334   {%
1335     \edef\@do@glssee{\noexpand\glsxtrindexseealso
1336       {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1337     \@do@glssee
1338   }%
1339 }%
1340 {%

```

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

```

1341   \ifdefvoid\@glo@see
1342   {%
1343     \edef\@do@glssee{\noexpand\glssee
1344       {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1345     \@do@glssee
1346   }%

```

```

1347     {}%
1348   }%
1349 }

```

```

1350 }

```

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

```

1351 \AtEndDocument{\if@glxtrindexcrossrefs\glxtraddallcrossrefs\fi}

```

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

```

1352 \newcommand*\glxtraddallcrossrefs{%
1353   \forallglossaries{\@glo@type}%
1354   {%
1355     \forglsentries[\@glo@type]{\@glo@label}%
1356     {%
1357       \ifglsused{\@glo@label}%
1358       {\expandafter\glxtr@addunusedxrefs\expandafter{\@glo@label}}}%
1359   }%
1360 }%
1361 }

```

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

```

1362 \newcommand*\@glxtr@addunusedxrefs[1]{%
1363   \letcs{\@glo@see}{glo\glstoklabel{#1}@see}%
1364   \ifdefvoid\@glo@see
1365   {}%
1366   {%
1367     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1368   }%
1369   \letcs{\@glo@see}{glo\glstoklabel{#1}@seealso}%
1370   \ifdefvoid\@glo@see
1371   {}%
1372   {%
1373     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1374   }%
1375 }

```

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

```

1376 \newcommand*\glxtr@addunused[1][ ]{%
1377   \@glxtr@addunused
1378 }

```

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

```

1379 \def\@glxtr@addunused#1\@end@glxtr@addunused{%
1380   \@for\@glxtr@label:=#1\do
1381   {%
1382     \ifglsused{\@glxtr@label}}}%

```

```

1383   {%
1384     \glsadd[format=glsxtrunusedformat]{\@glsxtr@label}%
1385     \glsunset{\@glsxtr@label}%
1386     \expandafter\@glsxtr@addunusedxrefs\expandafter{\@glsxtr@label}%
1387   }%
1388 }%
1389 }

```

xtrunusedformat

```

1390 \newcommand*{\glsxtrunusedformat}[1]{\unskip}

```

1.3.2 Document Definitions

`\ls@begindocdefs` This command was only introduced to glossaries v4.37, so it may not be defined. If it has been defined, redefine it to check `\@glsxtr@docdefval` so that it only inputs the `.glsdefs` file if `docdef=true`.

```

1391 \ifdef\gls@begindocdefs
1392 {%
1393   \renewcommand*{\gls@begindocdefs}{%
1394     \ifnum\@glsxtr@docdefval=1\relax
1395       \@gls@enablesavenonumberlist
1396       \edef\@gls@restoreat{%
1397         \noexpand\catcode'\noexpand\@=\number\catcode'\@ \relax}%
1398       \makeatletter
1399       \InputIfFileExists{\jobname.glsdefs}{\relax}%
1400       \@gls@restoreat
1401       \undef\@gls@restoreat
1402       \gls@defdocnewglossaryentry
1403     \fi
1404   }
1405 }
1406 {}

```

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

```

1407 \let\glsxtr@orgmakenoidxglossaries\makenoidxglossaries
1408 \renewcommand{\makenoidxglossaries}{%
1409   \ifdefequal\@glsxtr@record@setting\@glsxtr@record@setting@off
1410   {%
1411     \glsxtr@orgmakenoidxglossaries

```

Add marker to `\@do@seeglossary` but don't increment associated counter.

```

1412     \renewcommand{\@do@seeglossary}[2]{%
1413       \@@glsxtrwrglossmark
1414       \edef\@gls@label{\glsdetoklabel{##1}}%
1415       \protected@write\@auxout{\relax}{%
1416         \string\@gls@reference
1417         {\csname glo@\@gls@label @type\endcsname}%

```

```

1418      {\@gls@label}%
1419      {%
1420      \string\glsseeformat##2}%
1421      }%
1422    }%
1423  }%

```

Check for docdefs=restricted:

```

1424    \if@glxtrdocdefrestricted

```

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

```

1425      \renewcommand*{\@gls@reference}[3]{%
1426      \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
1427      \ifinlistcs{##2}{\glsref@##1}%
1428      }%
1429      {\listcsgadd{\glsref@##1}{##2}}%
1430      \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
1431      {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
1432      }%
1433      \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
1434    }%
1435  \else

```

Disable document definitions.

```

1436      \@glxtrdocdeffalse
1437      \fi
1438      \disable@keys{glossaries-extra}{docdef}%
1439    }%
1440    {%
1441      \PackageError{glossaries-extra}{\string\makenoidxglossaries\space
1442      not permitted\MessageBreak
1443      with record=\@glxtr@record@setting\space package option}%
1444      {You may only use \string\makenoidxglossaries\space with the
1445      record=off option}%
1446    }%
1447  }

```

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

```

1448 \renewcommand*{\@gls@defdocnewglossaryentry}{%
1449   \ifcase\@glxtr@docdefval
1450     docdef=false:
1451     \renewcommand*{\newglossaryentry}[2]{%
1452       \PackageError{glossaries-extra}{Glossary entries must
1453       be \MessageBreak defined in the preamble with \MessageBreak
1454       package option 'docdef=false'\MessageBreak(consider using
1455       'docdef=restricted')}{Move your glossary definitions to
1456       the preamble. You can also put them in a \MessageBreak separate file
1457       and load them with \string\loadglsentries.}%
1458     }%

```



```
1458 \or
      (docdef=true case.) Since the see value is now saved in a field, it can be used by entries that
      have been defined in the document.
```

```
1459 \let\gls@checkseeallowed\relax
1460 \let\newglossaryentry\new@glossaryentry
1461 \or
```

Restricted mode just needs to allow the see value.

```
1462 \let\gls@checkseeallowed\relax
1463 \fi
1464 }%
```

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`

```
1465 \newcommand*{\GlsXtrEnableOnTheFly}{%
1466   \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
1467 }
```

`\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.)

```
1468 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
1469   \renewcommand*{\glsdetoklabel}[1]{%
1470     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
1471     {%
1472       \expandafter\detokenize\expandafter{##1}%
1473     }%
1474     {\detokenize{##1}}}%
1475   }%
1476   \@GlsXtrEnableOnTheFly
1477 }
1478 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
1479   \expandafter\if\glsbackslash#1%
1480     #3%
1481   \else
1482     #4%
1483   \fi
1484 }
```

`\sXtrstarflywarn`

```
1485 \newcommand*{\glsxtrstarflywarn}{%
1486   \GlossariesExtraWarning{Experimental starred version of
1487   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
1488   read the warnings in the glossaries-extra user manual)}}%
1489 }
```

rEnableOnTheFly

```
1490 \newcommand*{\@GlsXtrEnableOnTheFly}{%
```

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

These definitions are all assigned the category given by:

\glsxtrcat

```
1491 \newcommand*{\glsxtrcat}{general}
```

\glsxtr

```
1492 \newcommand*{\glsxtr}[1] [] {%
1493   \def\glsxtr@keylist{##1}%
1494   \@glsxtr
1495 }
```

\@glsxtr

```
1496 \newcommand*{\@glsxtr}[2] [] {%
1497   \ifglentryexists{##2}%
1498   {%
1499     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1500   }%
1501   {%
1502     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1503       description={\nopostdesc},##1}%
1504   }%
1505   \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
1506 }
```

\Glsxtr

```
1507 \newcommand*{\Glsxtr}[1] [] {%
1508   \def\glsxtr@keylist{##1}%
1509   \@Glsxtr
1510 }
```

\@Glsxtr

```
1511 \newcommand*{\@Glsxtr}[2] [] {%
1512   \ifglentryexists{##2}%
1513   {%
1514     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1515   }%
1516   {%
1517     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1518       description={\nopostdesc},##1}%
1519   }%
1520   \expandafter\Gls\expandafter[\glsxtr@keylist]{##2}%
1521 }
```

\glsxtrpl

```
1522 \newcommand*\glsxtrpl[1][]{%
1523 \def\glsxtr@keylist{##1}%
1524 \@glsxtrpl
1525 }
```

\@glsxtrpl

```
1526 \newcommand*\@glsxtrpl[2][]{%
1527 \ifglsentryexists{##2}%
1528 {%
1529 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
1530 }%
1531 {%
1532 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1533 description={\nopostdesc},##1}%
1534 }%
1535 \expandafter\glspl\expandafter[\glsxtr@keylist]{##2}%
1536 }
```

\Glsxtrpl

```
1537 \newcommand*\Glsxtrpl[1][]{%
1538 \def\glsxtr@keylist{##1}%
1539 \@Glsxtrpl
1540 }
```

\@Glsxtrpl

```
1541 \newcommand*\@Glsxtrpl[2][]{%
1542 \ifglsentryexists{##2}
1543 {%
1544 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
1545 }%
1546 {%
1547 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1548 description={\nopostdesc},##1}%
1549 }%
1550 \expandafter\Glspl\expandafter[\glsxtr@keylist]{##2}%
1551 }
```

\GlsXtrWarning

```
1552 \newcommand*\GlsXtrWarning[2]{%
1553 \def\@glsxtr@optlist{##1}%
1554 \@onelevel@sanitize\@glsxtr@optlist
1555 \GlossariesExtraWarning{The options ‘\@glsxtr@optlist’ have
1556 been ignored for entry ‘##2’ as it has already been defined}%
1557 }
```

Disable commands after the glossary:

```
1558 \renewcommand\@printglossary[2]{%
```

```

1559 \def\@glxtr@printglossopts{##1}%
1560 \@glxtr@orgprintglossary{##1}{##2}%
1561 \def\@glxtr{\@glxtr@disabledflycommand\glxtr}%
1562 \def\@glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
1563 \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
1564 \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
1565 }

```

abledflycommand

```

1566 \newcommand*{\@glxtr@disabledflycommand}[1]{%
1567 \PackageError{glossaries-extra}%
1568 {\string##1\space can't be used after any of the \MessageBreak
1569 glossaries have been displayed}%
1570 {The on-the-fly commands enabled by
1571 \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1572 before the glossaries. If you want to use any entries \MessageBreak
1573 after any of the glossaries, you must use the standard \MessageBreak
1574 method of first defining the entry and then using the \MessageBreak
1575 entry with commands like \string\gls}%
1576 \@glxtr@disabledflycommand
1577 }%
1578 \newcommand*{\@glxtr@disabledflycommand}[2][\@glxtr@disabledflycommand]{##2}

```

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

```

1579 \let\GlsXtrEnableOnTheFly\relax
1580 }
1581 \@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.

```

1582 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}

```

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

etglossarystyle

```

1583 \renewcommand*{\setglossarystyle}[1]{%
1584 \ifcsundef{@glsstyle@#1}%
1585 {%
1586 \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%
1587 }%
1588 {%
1589 \csname @glsstyle@#1\endcsname

```

Only set the current style if it exists.

```

1590 \protected@edef\@glxtr@current@style{#1}%
1591 }%

```

```

1592 \ifx\@glossary@default@style\relax
1593   \protected@edef\@glossary@default@style{#1}%
1594 \fi
1595 }

```

In case we have an old version of glossaries:

```

1596 \ifdef\@glossary@default@style
1597 {}
1598 {%
1599   \let\@glossary@default@style\relax
1600 }

```

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

```

1601 \ifdef\glslistdottedwidth
1602 {%
1603   \ifdim\glslistdottedwidth=.5\hsize
1604     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1605     \AtBeginDocument{%
1606       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1607         \setlength{\glslistdottedwidth}{.5\columnwidth}%
1608       \fi
1609     }%
1610   \fi
1611 }
1612 {}%

```

Similarly for `\glsdescwidth`:

`\glsdescwidth`

```

1613 \ifdef\glsdescwidth
1614 {%
1615   \ifdim\glsdescwidth=.6\hsize
1616     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1617     \AtBeginDocument{%
1618       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1619         \setlength{\glsdescwidth}{.6\columnwidth}%
1620       \fi
1621     }%
1622   \fi
1623 }
1624 {}%

```

and for `\glspagelistwidth`:

`lspagelistwidth`

```

1625 \ifdef\glspagelistwidth
1626 {%
1627   \ifdim\glspagelistwidth=.1\hsize
1628     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}

```

```

1629 \AtBeginDocument{%
1630 \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1631 \setlength{\glspagelistwidth}{.1\columnwidth}%
1632 \fi
1633 }%
1634 \fi
1635 }
1636 {}%

```

aryentrynumbers Has the nonumberlist option been used?

```

1637 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1638 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1639 \glsnonumberlistfalse
1640 \renewcommand*{\glossaryentrynumbers}[1]{%
1641 \ifglentryexists{\glscurrententrylabel}%
1642 {%
1643 \@glxtrpreloctag
1644 \GlsXtrFormatLocationList{#1}%
1645 \@glxtrpostloctag
1646 \gls@save@numberlist{#1}%
1647 }{}%
1648 }%
1649 \else
1650 \glsnonumberlisttrue
1651 \renewcommand*{\glossaryentrynumbers}[1]{%
1652 \ifglentryexists{\glscurrententrylabel}%
1653 {%
1654 \gls@save@numberlist{#1}%
1655 }{}%
1656 }%
1657 \fi

```

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

```

1658 \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

```

1659 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
1660 \let\@glxtrpreloctag\@glxtrpreloctag
1661 \let\@glxtrpostloctag\@glxtrpostloctag
1662 \renewcommand*{\@glxtr@pagetag}{#1}%
1663 \renewcommand*{\@glxtr@pagetag}{#2}%
1664 \renewcommand*{\@glxtr@savepreloctag}[2]{%

```

```

1665 \csgdef{@glxtr@preloctag@##1}{##2}%
1666 }%
1667 \renewcommand*{\@glxtr@doloctag}{%
1668 \ifcsundef{@glxtr@preloctag@glscurrententrylabel}%
1669 {%
1670 \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
1671 Rerun required}%
1672 }%
1673 {%
1674 \csuse{@glxtr@preloctag@glscurrententrylabel}%
1675 }%
1676 }%
1677 }
1678 \@onlypreamble\GlsXtrEnablePreLocationTag

```

glxtrpreloctag

```

1679 \newcommand*{\@glxtrpreloctag}{%
1680 \let\@glxtr@org@delimN\delimN
1681 \let\@glxtr@org@delimR\delimR
1682 \let\@glxtr@org@glsignore\glsignore
1683 \gdef is required as the delimiters may occur inside a scope.
1684 \gdef \@glxtr@thisloctag{\@glxtr@pagetag}%
1685 \renewcommand*{\delimN}{%
1686 \gdef \@glxtr@thisloctag{\@glxtr@pagetag}%
1687 \@glxtr@org@delimN}%
1688 \renewcommand*{\delimR}{%
1689 \gdef \@glxtr@thisloctag{\@glxtr@pagetag}%
1690 \@glxtr@org@delimR}%
1691 \renewcommand*{\glsignore}[1]{%
1692 \gdef \@glxtr@thisloctag{\relax}%
1693 \@glxtr@org@glsignore{##1}}%
1694 }

```

glxtrpreloctag

```

1695 \newcommand*{\@glxtrpreloctag}{}

```

@glxtr@pagetag

```

1696 \newcommand*{\@glxtr@pagetag}{}%

```

glxtr@pagetag

```

1697 \newcommand*{\@glxtr@pagetag}{}%

```

lsxtrpostloctag

```

1698 \newcommand*{\@glxtrpostloctag}{%
1699 \let\delimN\@glxtr@org@delimN
1700 \let\delimR\@glxtr@org@delimR
1701 \let\glsignore\@glxtr@org@glsignore

```

```

1702 \protected@write\@auxout{}\%
1703 {\string\@glxtr@savepreloctag{\glscurrententrylabel}\@glxtr@thisloctag}}%
1704 }

```

lsxtrpostloctag

```

1705 \newcommand*\@glxtrpostloctag{}\%

```

lsxtr@preloctag

```

1706 \newcommand*\@glxtr@savepreloctag}[2]{%
1707 \protected@write\@auxout{}\%
1708 \string\providecommand\string\@glxtr@savepreloctag[2]{}}

```

glxtr@doloctag

```

1709 \newcommand*\@glxtr@doloctag{}\%

```

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

```

1710 \renewcommand*\KV@printgloss@nonumberlist}[1]{%
1711 \XKV@plfalse
1712 \XKV@sttrue
1713 \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
1714 {%
1715 \csname glsnonumberlist\XKV@resa\endcsname
1716 \ifglsnonumberlist
1717 \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1718 \else
1719 \def\glossaryentrynumbers##1{%
1720 \@glxtrpreloctag
1721 \GlsXtrFormatLocationList{##1}%
1722 \@glxtrpostloctag
1723 \gls@save@numberlist{##1}}%
1724 \fi
1725 }%
1726 }

```

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.

```

1727 \renewcommand*\glsentryfmt{%
1728 \ifglshasshort{\glslabel}\glssetabbrvfmt{\glscategory{\glslabel}}}%
1729 \glsifregular{\glslabel}%
1730 {\glxtrregularfont{\glsentryfmt}}%

```



```

1731 {%
1732   \ifglshasshort{\glslabel}%
1733   {\glsxtrabbreviationfont{\glsxtrgenabbrvfmt}}}%
1734   {\glsxtrregularfont{\glsgenentryfmt}}}%
1735 }%
1736 }

```

sxtrregularfont Font used for regular entries.

```
1737 \newcommand*{\glsxtrregularfont}[1]{#1}
```

bbreviationfont Font used for abbreviation entries.

```
1738 \newcommand*{\glsxtrabbreviationfont}[1]{#1}
```

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

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

```
1739 \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).

```

1740   \glsxtr@record{#2}{#3}{glslink}%
1741   \glsdoifexists{#3}%
1742   {%

```

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).

```

1743   \let\glxtrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
1744   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1745   \def\glscustomtext{#4}%
1746   \@glxtr@field@linkdefs
1747   #1%
1748   \@gls@link[#2]{#3}{#4}%
1749   \let\ifKV@glslink@hyper\glxtrorg@ifKV@glslink@hyper
1750   }%
1751   \glspostlinkhook
1752 }

```

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

\@gls@ Save the original definition and redefine.

```

1753 \let\@glxtr@org@gls@\@gls@
1754 \def\@gls@#1#2{%

```

```

1755 \@glstr@record{#1}{#2}{glslink}%
1756 \@glstr@org@glstr@{#1}{#2}%
1757 }%

```

`\@glsp1@` Save the original definition and redefine.

```

1758 \let\@glstr@org@glsp1@\@glsp1@
1759 \def\@glsp1@#1#2{%
1760 \@glstr@record{#1}{#2}{glslink}%
1761 \@glstr@org@glsp1@{#1}{#2}%
1762 }%

```

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

```

1763 \let\@glstr@org@Gls@\@Gls@
1764 \def\@Gls@#1#2{%
1765 \@glstr@record{#1}{#2}{glslink}%
1766 \@glstr@org@Gls@{#1}{#2}%
1767 }%

```

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

```

1768 \let\@glstr@org@Glspl@\@Glspl@
1769 \def\@Glspl@#1#2{%
1770 \@glstr@record{#1}{#2}{glslink}%
1771 \@glstr@org@Glspl@{#1}{#2}%
1772 }%

```

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

```

1773 \let\@glstr@org@GLS@\@GLS@
1774 \def\@GLS@#1#2{%
1775 \@glstr@record{#1}{#2}{glslink}%
1776 \@glstr@org@GLS@{#1}{#2}%
1777 }%

```

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

```

1778 \let\@glstr@org@GLSpl@\@GLSpl@
1779 \def\@GLSpl@#1#2{%
1780 \@glstr@record{#1}{#2}{glslink}%
1781 \@glstr@org@GLSpl@{#1}{#2}%
1782 }%

```

`\@glsdisp` This is redefined to allow the recording on the first run. Can't save and restore `\@glsdisp` since it has an optional argument.

```

1783 \renewcommand*{\@glsdisp}[3][{}]{%
1784 \@glstr@record{#1}{#2}{glslink}%
1785 \glsdoifexists{#2}{%
1786 \let\do@glstr@link@checkfirsthyper\@glstr@link@checkfirsthyper
1787 \let\glstr@ifplural\@secondoftwo
1788 \let\glstr@scapscase\@firstofthree
1789 \def\glstr@customtext{#3}%

```

```

1790 \def\glsinsert{}%
1791 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
1792 \@gls@link[#1]{#2}{\@glo@text}%
1793 \ifKV@glslink@local
1794 \glslocalunset{#2}%
1795 \else
1796 \glsunset{#2}%
1797 \fi
1798 }%
1799 \glspostlinkhook
1800 }

```

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

```

1801 \renewcommand*{\@gls@@link}[3][{}]{%
1802 \@glxtr@record{#1}{#2}{glslink}%
1803 \glsdoifexistsordo{#2}%
1804 {%
1805 \let\do@gls@link@checkfirsthyper\relax
1806 \@gls@link[#1]{#2}{#3}%
1807 }%
1808 {%
1809 \glstextformat{#3}%
1810 }%
1811 \glspostlinkhook
1812 }

```

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

```

1813 \newcommand*{\glxtrinitwrgloss}{%
1814 \glsifattribute{\glslabel}{wrgloss}{after}%
1815 {%
1816 \glxtrinitwrglossbeforefalse
1817 }%
1818 {%
1819 \glxtrinitwrglossbeforetrue
1820 }%
1821 }

```

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

```

1822 \newif\ifglxtrinitwrglossbefore
1823 \glxtrinitwrglossbeforetrue

```

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

```

1824 \define@choicekey{glslink}{wrgloss}%
1825 [\@glxtr@wrglossval\@glxtr@wrglossnr]%
1826 {before,after}%
1827 {%
1828 \ifcase\@glxtr@wrglossnr\relax
1829 \glxtrinitwrglossbeforetrue

```

```

1830 \or
1831 \glstrinitwrglossbeforefalse
1832 \fi
1833 }

1834 \define@key{glslink}{thevalue}{\def\@glstr@thevalue{#1}}

1835 \define@key{glslink}{theHvalue}{\def\@glstr@theHvalue{#1}}

tr@hyperoutside Define a hyperoutside key to determine whether \hyperlink should be outside \glstextformat.
1836 \define@boolkey{glslink}[glstr@]{hyperoutside}[true]{}
1837 \glstr@hyperoutsidetrue

ocal@textformat Provide a key to locally change the text format.
1838 \define@key{glslink}{textformat}{%
1839 \ifcsdef{#1}
1840 {%
1841 \letcs{\@glstr@local@textformat}{#1}%
1842 }%
1843 {%
1844 \PackageError{glossaries-extra}{Unknown control sequence name ‘#1’}{}%
1845 }%
1846 }

nithyperoutside Set the default if the hyperoutside is omitted.
1847 \newcommand*{\glstrnithyperoutside}{%
1848 \gl@ifattribute{\glslabel}{hyperoutside}{false}%
1849 {%
1850 \glstr@hyperoutsidetrue
1851 }%
1852 {%
1853 \glstr@hyperoutsidefalse
1854 }%
1855 }

r@inc@linkcount Does nothing by default.
1856 \newcommand*{\glstr@inc@linkcount}{}

linkpresetkeys User hook performed immediately before options are set. Does nothing by default.
1857 \newcommand*{\glslinkpresetkeys}{}

sXtrExpandedFmt Helper command that (protected) fully expands second argument and then applies it to the
first, which must be a command that takes a single argument.
1858 \newrobustcmd*{\GlsXtrExpandedFmt}[2]{%
1859 \protected@edef\@glstr@tmp{#2}%
1860 \expandafter#1\expandafter{\@glstr@tmp}%
1861 }

```

`\@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.

```

1862 \def\@gls@link[#1]#2#3{%
1863   \leavevmode
1864   \edef\glslabel{\glsdetoklabel{#2}}%
1865   \def\@gls@link@opts{#1}%
1866   \let\@gls@link@label\glslabel
1867   \let\@glsnumberformat\@glsxtr@defaultnumberformat
1868   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
1869   \edef\gls@type{\csname glo@\glslabel @type\endcsname}%
1870   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper

  Initialise \@glsxtr@local@textformat
1871   \let\@glsxtr@local@textformat\relax

  Initialise thevalue and theHvalue (v1.19).
1872   \def\@glsxtr@thevalue{%
1873     \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%

    Initialise when indexing should occur (new to v1.14).
1874    \glsxtr@nitr@wrgloss

    Initialise whether \hyperlink should be outside \gls@textformat (new to v1.21).
1875    \glsxtr@nitr@hyperoutside

    Note that the default link options may override \glsxtr@nitr@wrgloss.
1876    \@gls@setdefault@glslink@opts

    Increment link counter if enabled (new to v1.26).
1877    \glsxtr@inc@linkcount

    As the original definition.
1878    \do@gl@disablehyperinlist
1879    \do@gl@link@checkfirsthyper

    User hook before options are set (new to v1.26):
1880    \glslink@presetkeys

    Set options.
1881    \setkeys{glslink}{#1}%

    User hook after options are set:
1882    \glslink@postsetkeys

    Check thevalue and theHvalue before saving (v1.19).
1883    \ifdefempty{\@glsxtr@thevalue}%
1884      {%
1885        \@gls@saveentrycounter
1886      }%
1887      {%
1888        \let\theglsentrycounter\@glsxtr@thevalue
1889        \def\theHglentrycounter{\@glsxtr@theHvalue}%

```

```

1890 }%
1891 \@gls@setsort{\glslabel}%

Check if the textformat key has been used.
1892 \ifx\@glsxtr@local@textformat\relax

Check textformat attribute (new to v1.21).
1893 \gls@hasattribute{\glslabel}{textformat}%
1894 {%
1895 \edef\@glsxtr@attrval{\gls@getattribute{\glslabel}{textformat}}%
1896 \ifcsdef{\@glsxtr@attrval}%
1897 {%
1898 \letcs{\@glsxtr@textformat}{\@glsxtr@attrval}%
1899 }%
1900 {%
1901 \GlossariesExtraWarning{Unknown control sequence name
1902 '\@glsxtr@attrval' supplied in textformat attribute
1903 for entry '\glslabel'. Reverting to default \string\glstextformat}%
1904 \let\@glsxtr@textformat\glstextformat
1905 }%
1906 }%
1907 {%
1908 \let\@glsxtr@textformat\glstextformat
1909 }%
1910 \else
1911 \let\@glsxtr@textformat\@glsxtr@local@textformat
1912 \fi

Do write if it should occur before the link text:
1913 \ifglsxtr@nitrinitwrglossbefore
1914 \@do@wrglossary{#2}%
1915 \fi

Do the link text:
1916 \ifKV@glslink@hyper
1917 \ifglsxtr@hyperoutside
1918 \@glslink{\glslinkprefix\glslabel}{\@glsxtr@textformat{#3}}%
1919 \else
1920 \@glsxtr@textformat{\@glslink{\glslinkprefix\glslabel}{#3}}%
1921 \fi
1922 \else
1923 \ifglsxtr@hyperoutside
1924 \glsdonohyperlink{\glslinkprefix\glslabel}{\@glsxtr@textformat{#3}}%
1925 \else
1926 \@glsxtr@textformat{\glsdonohyperlink{\glslinkprefix\glslabel}{#3}}%
1927 \fi
1928 \fi

Do write if it should occur after the link text:
1929 \ifglsxtr@nitrinitwrglossbefore
1930 \else

```

```

1931 \do@wrglossary{#2}%
1932 \fi

As the original definition:
1933 \let@ifKV@glslink@hyper\org@ifKV@glslink@hyper
1934 }

1935 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}

1936 \define@key{glossadd}{theHvalue}{\def\@glxtr@theHvalue{#1}}

lsaddpresetkeys
1937 \newcommand*{\gl saddpresetkeys}{ }

saddpostsetkeys
1938 \newcommand*{\gl saddpostsetkeys}{ }

\gl sadd Redefine to include \@glxtr@record and suppress in headings
1939 \renewrobustcmd*{\gl sadd}[2][ ]{%
1940 \glxtrifinmark
1941 }{%
1942 {%
1943 \@gl s@adjustmode
1944 \@glxtr@record{#1}{#2}{glossadd}%
1945 \gl sdoifexists{#2}%
1946 {%
1947 \let\@gl snumberformat\@glxtr@defaultnumberformat
1948 \edef\@gl s@counter{\csname glo\@gl sdetoklabel{#2}@counter\endcsname}%
1949 \def\@glxtr@thevalue{%
1950 \def\@glxtr@theHvalue{\@glxtr@thevalue}%

Implement any default settings (before options are set)
1951 \gl saddpresetkeys
1952 \setkeys{glossadd}{#1}%

Implement any default settings (after options are set)
1953 \gl saddpostsetkeys
1954 \ifdefempty{\@glxtr@thevalue}%
1955 {%
1956 \@gl s@saveentrycounter
1957 }%
1958 {%
1959 \let\thegl sentrycounter\@glxtr@thevalue
1960 \def\theHgl sentrycounter{\@glxtr@theHvalue}%
1961 }%

Define sort key if necessary (in case of sort=use):
1962 \@gl s@setsort{#2}%
1963 \@do@wrglossary{#2}%
1964 }%
1965 }%
1966 }

```

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

```
1967 \newcommand*{\@glsxtr@field@linkdefs}{%
1968   \let\glsxtrifwasfirstuse\@secondoftwo
1969   \let\glsifplural\@secondoftwo
1970   \let\glscapscase\@firstofthree
1971   \let\glsinsert\@empty
1972 }
```

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

assignfieldfont

```
1973 \newcommand*{\glsxtrassignfieldfont}[1]{%
1974   \ifglstryexists{#1}%
1975   {%
1976     \ifglshasshort{#1}%
1977     {%
1978       \glssetabbrvfmt{\glscategory{#1}}%
1979       \glsifregular{#1}%
1980       {\let\@gls@field@font\glsxtrregularfont}%
1981       {\let\@gls@field@font\@firstofone}%
1982     }%
1983     {%
1984       \glsifnotregular{#1}%
1985       {\let\@gls@field@font\@firstofone}%
1986       {\let\@gls@field@font\glsxtrregularfont}%
1987     }%
1988   }%
1989   {%
1990     \let\@gls@field@font\@gobble
1991   }%
1992 }
```

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

```
1993 \def\@glstext@#1#2[#3]{%
1994   \glsxtrassignfieldfont{#2}%
1995   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
1996 }
```

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

```
1997 \def\@GLStext@#1#2[#3]{%
1998   \glsxtrassignfieldfont{#2}%
1999   \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
2000   {\@gls@field@font{\@GLSaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
2001 }
```

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

```
2002 \def\@Glstext@#1#2[#3]{%
2003   \glsxtrassignfieldfont{#2}%

```



```

2004 \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
2005 {\@gls@field@font{\Glsaccessstext{#2}#3}}%
2006 }

```

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`

```

2007 \newcommand*{\glstrchecknohyperfirst}[1]{%
2008 \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
2009 }

```

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

```

2010 \def\@glsfirst@#1#2[#3]{%
2011 \glstrassignfieldfont{#2}%
    Ensure that \glsfirst honours the nohyperfirst attribute.
2012 \@gls@field@link
2013 [\let\glstrifwasfirstuse\@firstoftwo
2014 \glstrchecknohyperfirst{#2}%
2015 ]{#1}{#2}%
2016 {\@gls@field@font{\glsaccessfirst{#2}#3}}%
2017 }

```

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

```

2018 \def\@Glsfirst@#1#2[#3]{%
2019 \glstrassignfieldfont{#2}%
    Ensure that \Glsfirst honours the nohyperfirst attribute.
2020 \@gls@field@link
2021 [\let\glstrifwasfirstuse\@firstoftwo
2022 \let\glscapscase\@secondofthree
2023 \glstrchecknohyperfirst{#2}%
2024 ]%
2025 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
2026 }

```

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

```

2027 \def\@GLSfirst@#1#2[#3]{%
2028 \glstrassignfieldfont{#2}%
    Ensure that \GLSfirst honours the nohyperfirst attribute.
2029 \@gls@field@link
2030 [\let\glstrifwasfirstuse\@firstoftwo
2031 \let\glscapscase\@thirdofthree
2032 \glstrchecknohyperfirst{#2}%
2033 ]%
2034 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
2035 }

```

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

```
2036 \def\@glsplural@#1#2[#3]{%
2037   \glsxtrassignfieldfont{#2}%
2038   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
2039   {\@gls@field@font{\glsaccessplural{#2}#3}}%
2040 }
```

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

```
2041 \def\@Glsplural@#1#2[#3]{%
2042   \glsxtrassignfieldfont{#2}%
2043   \@gls@field@link
2044   [\let\glsifplural\@firstoftwo
2045    \let\glscapscase\@secondofthree
2046   ]%
2047   {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
2048 }
```

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

```
2049 \def\@GLSplural@#1#2[#3]{%
2050   \glsxtrassignfieldfont{#2}%
2051   \@gls@field@link
2052   [\let\glsifplural\@firstoftwo
2053    \let\glscapscase\@thirdofthree
2054   ]%
2055   {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
2056 }
```

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

```
2057 \def\glsfirstplural@#1#2[#3]{%
2058   \glsxtrassignfieldfont{#2}%
2059   \@gls@field@link
2060   [\let\glsxtrifwasfirstuse\@firstoftwo
2061    \let\glsifplural\@firstoftwo
2062    \glsxtrchecknohyperfirst{#2}%
2063   ]%
2064   {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
2065 }
```

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

```
2066 \def\Glsfirstplural@#1#2[#3]{%
2067   \glsxtrassignfieldfont{#2}%
2068   \@gls@field@link
2069   [\let\glsxtrifwasfirstuse\@firstoftwo
2070    \let\glsifplural\@firstoftwo
2071    \let\glscapscase\@secondofthree
```

```

2072 \glstrchecknohyperfirst{#2}%
2073 ]%
2074 {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
2075 }

```

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

```

2076 \def\@GLSfirstplural@#1#2[#3]{%
2077 \glstrassignfieldfont{#2}%
    Ensure that \glsfirstplural honours the nohyperfirst attribute.
2078 \@gls@field@link
2079 [\let\glstrifwasfirstuse\@firstoftwo
2080 \let\glsifplural\@firstoftwo
2081 \let\glscapscase\@thirdofthree
2082 \glstrchecknohyperfirst{#2}%
2083 ]%
2084 {#1}{#2}%
2085 {\@gls@field@font{\Glsaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
2086 }

```

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

```

2087 \def\@glsname@#1#2[#3]{%
2088 \glstrassignfieldfont{#2}%
2089 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
2090 }

```

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

```

2091 \def\@Glsname@#1#2[#3]{%
2092 \glstrassignfieldfont{#2}%
2093 \@gls@field@link
2094 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2095 {\@gls@field@font{\Glsaccessname{#2}#3}}%
2096 }

```

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

```

2097 \def\@GLSname@#1#2[#3]{%
2098 \glstrassignfieldfont{#2}%
2099 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2100 {#1}{#2}%
2101 {\@gls@field@font{\Glsaccessname{#2}\mfirstucMakeUppercase{#3}}}%
2102 }

```

\@glsdesc@

```

2103 \def\@glsdesc@#1#2[#3]{%
2104 \glstrassignfieldfont{#2}%
2105 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
2106 }

```

```

\@Glsdesc@ First letter uppercase version.
2107 \def\@Glsdesc@#1#2[#3]{%
2108   \glstrassignfieldfont{#2}%
2109   \@gls@field@link
2110   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2111   {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
2112 }

\@GLSdesc@ All uppercase version.
2113 \def\@GLSdesc@#1#2[#3]{%
2114   \glstrassignfieldfont{#2}%
2115   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2116   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2117 }

@glsglscapscase@ No case-changing version.
2118 \def\@glsglscapscase@#1#2[#3]{%
2119   \glstrassignfieldfont{#2}%
2120   \@gls@field@link
2121   [\let\glscapscase\@secondoftwo
2122   \let\glsifplural\@firstoftwo
2123   ]{#1}{#2}{\@gls@field@font{\glsglscapscase{#2}#3}}%
2124 }

@Glsdescplural@ First letter uppercase version.
2125 \def\@Glsdescplural@#1#2[#3]{%
2126   \glstrassignfieldfont{#2}%
2127   \@gls@field@link
2128   [\let\glscapscase\@secondoftwo
2129   \let\glsifplural\@firstoftwo
2130   ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
2131 }

@GLSdescplural@ All uppercase version.
2132 \def\@GLSdescplural@#1#2[#3]{%
2133   \glstrassignfieldfont{#2}%
2134   \@gls@field@link
2135   [\let\glscapscase\@thirdoftwo
2136   \let\glsifplural\@firstoftwo
2137   ]%
2138   {#1}{#2}%
2139   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
2140 }

\@glssymbol@
2141 \def\@glssymbol@#1#2[#3]{%
2142   \glstrassignfieldfont{#2}%
2143   \@gls@field@link{#1}{#2}{\@gls@field@font{\glssymbol{#2}#3}}%
2144 }

```

\@Glssymbol@ First letter uppercase version.

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

\@GLSsymbol@ All uppercase version.

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

lssymbolplural@ No case-changing version.

```
2156 \def\@lssymbolplural@#1#2[#3]{%
2157   \glstrassignfieldfont{#2}%
2158   \@gls@field@link
2159   [\let\glscapscase\@secondoftwo
2160   \let\glsifplural\@firstoftwo
2161   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2162 }
```

lssymbolplural@ First letter uppercase version.

```
2163 \def\@lssymbolplural@#1#2[#3]{%
2164   \glstrassignfieldfont{#2}%
2165   \@gls@field@link
2166   [\let\glscapscase\@secondoftwo
2167   \let\glsifplural\@firstoftwo
2168   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2169 }
```

LSsymbolplural@ All uppercase version.

```
2170 \def\@LSsymbol@#1#2[#3]{%
2171   \glstrassignfieldfont{#2}%
2172   \@gls@field@link
2173   [\let\glscapscase\@thirdoftwo
2174   \let\glsifplural\@firstoftwo
2175   ]%
2176   {#1}{#2}%
2177   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2178 }
```

\@Glsuseri@ First letter uppercase version.

```
2179 \def\@Glsuseri@#1#2[#3]{%
2180   \glstrassignfieldfont{#2}%
2181   \@gls@field@link
```

```

2182 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2183 {\@gls@field@font{\Glsentryuseri{#2}#3}}%
2184 }

```

\@GLSuseri@ All uppercase version.

```

2185 \def\@GLSuseri@#1#2[#3]{%
2186 \glstrassignfieldfont{#2}%
2187 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2188 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseri{#2}#3}}}%
2189 }

```

\@Glsuserii@ First letter uppercase version.

```

2190 \def\@Glsuserii@#1#2[#3]{%
2191 \glstrassignfieldfont{#2}%
2192 \@gls@field@link
2193 [\let\glscapscase\@secondoftwo]%
2194 {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
2195 }

```

\@GLSuserii@ All uppercase version.

```

2196 \def\@GLSuserii@#1#2[#3]{%
2197 \glstrassignfieldfont{#2}%
2198 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2199 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
2200 }

```

\@Glsuseriii@ First letter uppercase version.

```

2201 \def\@Glsuseriii@#1#2[#3]{%
2202 \glstrassignfieldfont{#2}%
2203 \@gls@field@link
2204 [\let\glscapscase\@secondoftwo]%
2205 {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
2206 }

```

\@GLSuseriii@ All uppercase version.

```

2207 \def\@GLSuseriii@#1#2[#3]{%
2208 \glstrassignfieldfont{#2}%
2209 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2210 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
2211 }

```

\@Glsuseriv@ First letter uppercase version.

```

2212 \def\@Glsuseriv@#1#2[#3]{%
2213 \glstrassignfieldfont{#2}%
2214 \@gls@field@link
2215 [\let\glscapscase\@secondoftwo]%
2216 {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
2217 }

```

\@GLSuseriv@ All uppercase version.

```
2218 \def\@GLSuseriv@#1#2[#3]{%
2219   \glstrassignfieldfont{#2}%
2220   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2221   {#1}{#2}%
2222   {\@gls@field@font{\mfirstucMakeUppercase{\glstentryuseriv{#2}#3}}}%
2223 }
```

\@Glsuserv@ First letter uppercase version.

```
2224 \def\@Glsuserv@#1#2[#3]{%
2225   \glstrassignfieldfont{#2}%
2226   \@gls@field@link
2227   [\let\glscapscase\@secondoftwo]%
2228   {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
2229 }
```

\@GLSuserv@ All uppercase version.

```
2230 \def\@GLSuserv@#1#2[#3]{%
2231   \glstrassignfieldfont{#2}%
2232   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2233   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glstentryuserv{#2}#3}}}%
2234 }
```

\@Glsuservi@ First letter uppercase version.

```
2235 \def\@Glsuservi@#1#2[#3]{%
2236   \glstrassignfieldfont{#2}%
2237   \@gls@field@link
2238   [\let\glscapscase\@secondoftwo]%
2239   {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}%
2240 }
```

\@GLSuservi@ All uppercase version.

```
2241 \def\@GLSuservi@#1#2[#3]{%
2242   \glstrassignfieldfont{#2}%
2243   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2244   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glstentryuservi{#2}#3}}}%
2245 }
```

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

\@acrshort No case change.

```
2246 \def\@acrshort#1#2[#3]{%
2247   \glsdoifexists{#2}%
2248   {%
2249     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2250     \let\glstrifwasfirstuse\@secondoftwo
2251     \let\glsifplural\@secondoftwo
```

```

2252 \let\glscapscase\@firstofthree
2253 \let\glsinsert\@empty
2254 \def\glscustomtext{%
2255     \acronymfont{\glsaccesssshort{#2}}#3%
2256 }%
2257 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2258 }%
2259 \glspostlinkhook
2260 }

```

\@Acrshort First letter uppercase.

```

2261 \def\@Acrshort#1#2[#3]{%
2262     \glsdoifexists{#2}%
2263     {%
2264         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2265         \let\glstrifwasfirstuse\@secondoftwo
2266         \let\glsifplural\@secondoftwo
2267         \let\glscapscase\@secondofthree
2268         \let\glsinsert\@empty
2269         \def\glscustomtext{%
2270             \acronymfont{\Glsaccesssshort{#2}}#3%
2271         }%
2272         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2273     }%
2274     \glspostlinkhook
2275 }

```

\@ACRshort All uppercase.

```

2276 \def\@ACRshort#1#2[#3]{%
2277     \glsdoifexists{#2}%
2278     {%
2279         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2280         \let\glstrifwasfirstuse\@secondoftwo
2281         \let\glsifplural\@secondoftwo
2282         \let\glscapscase\@thirdofthree
2283         \let\glsinsert\@empty
2284         \def\glscustomtext{%
2285             \mfirstucMakeUppercase{\acronymfont{\glsaccesssshort{#2}}#3}%
2286         }%
2287         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2288     }%
2289     \glspostlinkhook
2290 }

```

\@acrshortpl No case change.

```

2291 \def\@acrshortpl#1#2[#3]{%
2292     \glsdoifexists{#2}%
2293     {%
2294         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```



```

2295 \let\glxtrifwasfirstuse\@secondoftwo
2296 \let\glsifplural\@firstoftwo
2297 \let\glscapscase\@firstofthree
2298 \let\glsinsert\@empty
2299 \def\glscustomtext{%
2300 \acronymfont{\glsaccessshortpl{#2}}#3%
2301 }%
2302 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2303 }%
2304 \glspostlinkhook
2305 }

```

\@Acrshortpl First letter uppercase.

```

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

```

\@ACRshortpl All uppercase.

```

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

```

\@acrlong No case change.

```

2336 \def\@acrlong#1#2[#3]{%
2337 \glsdoifexists{#2}%

```

```

2338 {%
2339   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2340   \let\glxtrifwasfirstuse\@secondoftwo
2341   \let\gl@ifplural\@secondoftwo
2342   \let\gl@scapscase\@firstofthree
2343   \let\gl@insert\@empty
2344   \def\gl@customtext{%
2345     \acronymfont{\gl@saccesslong{#2}}#3%
2346   }%
2347   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2348 }%
2349 \glspostlinkhook
2350 }

```

\@Acrlong First letter uppercase.

```

2351 \def\@Acrlong#1#2[#3]{%
2352   \gl@doifexists{#2}%
2353   {%
2354     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2355     \let\glxtrifwasfirstuse\@secondoftwo
2356     \let\gl@ifplural\@secondoftwo
2357     \let\gl@scapscase\@secondofthree
2358     \let\gl@insert\@empty
2359     \def\gl@customtext{%
2360       \acronymfont{\gl@saccesslong{#2}}#3%
2361     }%
2362     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2363   }%
2364   \glspostlinkhook
2365 }

```

\@ACRlong All uppercase.

```

2366 \def\@ACRlong#1#2[#3]{%
2367   \gl@doifexists{#2}%
2368   {%
2369     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2370     \let\glxtrifwasfirstuse\@secondoftwo
2371     \let\gl@ifplural\@secondoftwo
2372     \let\gl@scapscase\@thirdofthree
2373     \let\gl@insert\@empty
2374     \def\gl@customtext{%
2375       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslong{#2}}#3}%
2376     }%
2377     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2378   }%
2379   \glspostlinkhook
2380 }

```

\@acrlongpl No case change.

```

2381 \def\@acrlongpl#1#2[#3]{%
2382   \glsdoifexists{#2}%
2383   {%
2384     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2385     \let\glxtrifwasfirstuse\@secondoftwo
2386     \let\gl@sifplural\@firstoftwo
2387     \let\glscapscase\@firstofthree
2388     \let\gl$insert\@empty
2389     \def\glscustomtext{%
2390       \acronymfont{\gl@saccesslongpl{#2}}#3%
2391     }%
2392     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2393   }%
2394   \glspostlinkhook
2395 }

```

\@Acrlongpl First letter uppercase.

```

2396 \def\@Acrlongpl#1#2[#3]{%
2397   \glsdoifexists{#2}%
2398   {%
2399     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2400     \let\glxtrifwasfirstuse\@secondoftwo
2401     \let\gl@sifplural\@firstoftwo
2402     \let\glscapscase\@secondofthree
2403     \let\gl$insert\@empty
2404     \def\glscustomtext{%
2405       \acronymfont{\Glsaccesslongpl{#2}}#3%
2406     }%
2407     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2408   }%
2409   \glspostlinkhook
2410 }

```

\@ACrlongpl All uppercase.

```

2411 \def\@ACrlongpl#1#2[#3]{%
2412   \glsdoifexists{#2}%
2413   {%
2414     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2415     \let\glxtrifwasfirstuse\@secondoftwo
2416     \let\gl@sifplural\@firstoftwo
2417     \let\glscapscase\@thirdofthree
2418     \let\gl$insert\@empty
2419     \def\glscustomtext{%
2420       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslongpl{#2}}#3}%
2421     }%
2422     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2423   }%
2424   \glspostlinkhook
2425 }

```

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

\@glsaddkey

```

2426 \renewcommand*{\@glsaddkey}[7]{%
2427   \key@ifundefined{glossentry}{#1}%
2428   {%
2429     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
2430     \appto\@gls@keymap{,{#1}{#1}}%
2431     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
2432     \appto\@newglossaryentryposthook{%
2433       \letcs{\@glo@tmp}{@glo@#1}%
2434       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
2435     }%
2436     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
2437     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

2438   \ifcsdef{@gls@user@#1@}%
2439   {%
2440     \PackageError{glossaries}%
2441     {Can't define '\string#5' as helper command
2442     '\expandafter\string\csname @gls@user@#1@\endcsname' already
2443     exists}%
2444     {}%
2445   }%
2446   {%
2447     \expandafter\newcommand\expandafter*\expandafter
2448     {\csname @gls@user@#1\endcsname}[2][ ]{%
2449       \new@ifnextchar[%
2450         {\csuse{@gls@user@#1@}{##1}{##2}}%
2451         {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
2452     \csdef{@gls@user@#1@}##1##2[##3]{%
2453       \@gls@field@link{##1}{##2}{#3{##2}##3}%
2454     }%
2455     \newrobustcmd*{#5}{%
2456       \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
2457   }%

```

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

```

2458   \ifcsdef{@Gls@user@#1@}%
2459   {%
2460     \PackageError{glossaries}%
2461     {Can't define '\string#6' as helper command
2462     '\expandafter\string\csname @Gls@user@#1@\endcsname' already
2463     exists}%
2464     {}%
2465   }%
2466   {%
2467     \expandafter\newcommand\expandafter*\expandafter

```

```

2468     {\csname @Gls@user@#1\endcsname}[2] [] {%
2469         \new@ifnextchar[%
2470             {\csuse{@Gls@user@#1@}{##1}{##2}}}%
2471             {\csuse{@Gls@user@#1@}{##1}{##2} [] }}}%
2472     \csdef{@Gls@user@#1@}##1##2[##3]{%
2473         \@gls@field@link[\let\glscapscase\secondofthree]%
2474         {##1}{##2}{#4{##2}##3}%
2475     }%
2476     \newrobustcmd*{#6}{%
2477         \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
2478     }%

```

Finally the all caps version (modified):

```

2479     \ifcsdef{@GLS@user@#1@}%
2480     {%
2481         \PackageError{glossaries}%
2482         {Can't define '\string#7' as helper command
2483         '\expandafter\string\csname @GLS@user@#1\endcsname' already
2484         exists}%
2485     }%
2486 }%
2487 {%
2488     \expandafter\newcommand\expandafter*\expandafter
2489     {\csname @GLS@user@#1\endcsname}[2] [] {%
2490         \new@ifnextchar[%
2491             {\csuse{@GLS@user@#1@}{##1}{##2}}}%
2492             {\csuse{@GLS@user@#1@}{##1}{##2} [] }}}%
2493     \csdef{@GLS@user@#1@}##1##2[##3]{%
2494         \@gls@field@link[\let\glscapscase\thirdofthree]%
2495         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
2496     }%
2497     \newrobustcmd*{#7}{%
2498         \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
2499     }%
2500 }%
2501 {%
2502     \PackageError{glossaries-extra}{Key '#1' already exists}{}%
2503 }%
2504 }

```

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

```

2505 \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.

```

2506 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
2507 \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.

```
2508 \ifglused{\glslabel}%
2509 {\let\glstrifwasfirstuse\@secondoftwo}
2510 {\let\glstrifwasfirstuse\@firstoftwo}%
```

Store the category label for convenience.

```
2511 \edef\glscategorylabel{\glscategory{\glslabel}}%
2512 \ifglused{\glslabel}%
2513 {%
2514   \glscategoryattribute{\glscategorylabel}{nohypernext}{true}%
2515   {\KV@glslink@hyperfalse}{}}%
2516 }%
2517 {%
2518   \glscategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
2519   {\KV@glslink@hyperfalse}{}}%
2520 }%
2521 \glslinkcheckfirsthyperhook
2522 }
```

`\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.

```
2523 \ifdefined\do@glsglsdisablehyperinlist
2524 {%
2525   \let\@glstr@do@glsglsdisablehyperinlist\do@glsglsdisablehyperinlist
2526   \renewcommand*{\do@glsglsdisablehyperinlist}{%
2527     \@glstr@do@glsglsdisablehyperinlist
2528     \glscategoryattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}}%
2529   }
2530 }
2531 {}
```

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

```
2532 \define@boolkey{glslink}{noindex}[true]{}
2533 \KV@glslink@noindexfalse
```

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

`\@glslink@opts`

```
2534 \ifdefined\@gls@setdefault@glslink@opts
2535 {
2536   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2537     \KV@glslink@noindexfalse
2538     \@glstrsetaliasnoindex
2539   }
2540 }
2541 {
```

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

```
2542 \newcommand*{\@glslsetdefault@glslink@opts}{%
2543   \KV@glslink@noindexfalse
2544   \@glsxtrsetaliasnoindex
2545 }
2546 \preto\do@glstdisablehyperinlist{\@glslsetdefault@glslink@opts}
2547 }
```

`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.)

```
2548 \providecommand*{\glsxtrsetaliasnoindex}{%
2549   \KV@glslink@noindextrue
2550 }
```

`setaliasnoindex`

```
2551 \newcommand*{\@glsxtrsetaliasnoindex}{%
2552   \glsxtrifhasfield{alias}{\glslabel}%
2553   {%
2554     \let\glsxtrindexaliased\@glsxtrindexaliased
2555     \glsxtrsetaliasnoindex
2556     \let\glsxtrindexaliased\@no@glsxtrindexaliased
2557   }%
2558   }%
2559 }
```

`xtrindexaliased`

```
2560 \newcommand{\@glsxtrindexaliased}{%
2561   \ifKV@glslink@noindex
2562   \else
2563     \begingroup
2564     \let\@glslnumberformat\@glsxtr@defaultnumberformat
2565     \edef\@glsl@counter{\csname glo@\glstdetoklabel{\glslabel}@counter\endcsname}%
2566     \glsxtr@saveentrycounter
2567     \@@do@wrglossary{\glsxtralias{\glslabel}}%
2568     \endgroup
2569   \fi
2570 }
```

`xtrindexaliased`

```
2571 \newcommand{\@no@glsxtrindexaliased}{%
2572   \PackageError{glossaries-extra}{\string\glsxtrindexaliased\space
2573     not permitted outside definition of \string\glsxtrsetaliasnoindex}%
2574   }%
2575 }
```

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

```
2576 \let\glsxtrindexaliased\@no@glsxtrindexaliased
```

DefaultGlsOpts Set the default options for \glslink etc.

```
2577 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
2578   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2579     \setkeys{glslink}{#1}%
2580     \@glsxtrsetaliasnoindex
2581   }%
2582 }
```

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

```
2583 \newcommand*{\glsxtrifindexing}[2]{%
2584   \ifKV@glslink@noindex #2\else #1\fi
2585 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```
2586 \renewcommand*{\glswriteentry}[2]{%
2587   \glsxtrifindexing
2588   {%
2589     \ifglsindexonlyfirst
2590       \ifglsused{#1}
2591       {\glsxtrdoautoindexname{#1}{dualindex}}%
2592       {#2}%
2593     \else
2594       \glsifattribute{#1}{indexonlyfirst}{true}%
2595       {\ifglsused{#1}
2596        {\glsxtrdoautoindexname{#1}{dualindex}}%
2597        {#2}}%
2598       {#2}%
2599     \fi
2600   }%
2601   {}%
2602 }
```

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

```
2603 \appto\@do@@wrglossary{\@glsxtr@do@@wrindex
2604   \glsxtrdowrglossaryhook{\@gls@label}%
2605 }
```

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

Similarly for the “noidx” version:

s@noidxglossary

```
2606 \appto\gls@noidxglossary{\@glsxtr@do@@wrindex
2607   \glsxtrdowrglossaryhook{\@gls@label}%
2608 }
```

xtr@do@@wrindex

```
2609 \newcommand*{\@glsxtr@do@@wrindex}{%
```


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

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

alt@glsl@hyp@opt User version

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

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

rSetAltModifier

org@dohyperlink

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

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

```
2637 \ifdef\glsnavhyperlink
2638 {
2639   \renewcommand*{\glsnavhyperlink}[3][\@glo@type]{%
2640     \edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%
```

Scope:

```
2641   {%
2642     \let\glsdohyperlink\glsxtr@org@dohyperlink
2643     \@glslink{\glsnavhyperlinkname{#1}{#2}}{#3}%
2644   }%
2645 }%
2646 }
2647 {}
```

`\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.

```
2648 \renewcommand*{\glsdohyperlink}[2]{%
2649   \glsasattribute{\glslabel}{targeturl}%
2650   {%
2651     \glsasattribute{\glslabel}{targetname}%
2652     {%
2653       \glsasattribute{\glslabel}{targetcategory}%
2654       {%
2655         \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2656           {\glsgetattribute{\glslabel}{targetcategory}}%
2657           {\glsgetattribute{\glslabel}{targetname}}%
2658           {\glsxtrprotectlinks#2}}%
2659       }%
2660     }%
2661     \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2662       }%
2663       {\glsgetattribute{\glslabel}{targetname}}%
2664       {\glsxtrprotectlinks#2}}%
2665   }%
2666 }%
2667 {%
2668   \href{\glsgetattribute{\glslabel}{targeturl}}{%
2669     {\glsxtrprotectlinks#2}}%
2670 }%
2671 }%
2672 {}
```

Check for alias.

```

2673 \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
2674 \ifdefvoid\gloaliaslabel
2675 {%
2676   \glsxtrhyperlink{#1}{\glsxtrprotectlinks#2}}%
2677 }%
2678 {%

```

Redirect link to the alias target.

```

2679   \glsxtrhyperlink
2680   {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
2681   {\glsxtrprotectlinks#2}}%
2682 }%
2683 }%
2684 }

```

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

```

2685 \ifdef\@glsshowtarget
2686 {
2687   \newcommand{\glsxtrhyperlink}[2]{%
2688     \@glsshowtarget{#1}%
2689     \hyperlink{#1}{#2}%
2690   }%
2691 }
2692 {
2693   \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%
2694 }

```

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.

```

2695 \renewrobustcmd*{\glsdohyperlink}[2][\glssentrytext{\@glo@label}]{%
2696   \glsdoifexists{#2}%
2697   {%
2698     \def\@glo@label{#2}%
2699     {\edef\glslabel{#2}%
2700      \@glslink{\glolinkprefix\glslabel}{#1}}%
2701   }%
2702 }

```

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

```

2703 \renewcommand{\glsdisablehyper}{%
2704   \KV@glslink@hyperfalse
2705   \def\@glslink{\glsdohyperlink}%
2706   \let\@glstarget\@secondoftwo
2707 }

```

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

```
2708 \renewcommand{\glsenablehyper}{%
2709 \KV@glslink@hypertrue
2710 \def\@glslink{\glsdohyperlink}%
2711 \def\@glstarget{\glsdohypertarget}%
2712 }
```

`\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.

```
2713 \def\glsdonohyperlink#1#2{\@glsxtrprotectlinks #2}}
```

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

```
2714 \ifcsundef{hyperlink}%
2715 {%
2716 \def\@glslink{\glsdonohyperlink}
2717 }%
2718 {%
2719 \def\@glslink{\glsdohyperlink}
2720 }
```

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

```
2721 \newcommand*{\glsxtrprotectlinks}{%
2722 \KV@glslink@hyperfalse
2723 \KV@glslink@noindextrue
2724 \let\@gls@\@glsxtr@p@text@
2725 \let\@Gls@\@Glsxtr@p@text@
2726 \let\@GLS@\@GLSxtr@p@text@
2727 \let\@glspl@\@glsxtr@p@plural@
2728 \let\@Glspl@\@Glsxtr@p@plural@
2729 \let\@GLSpl@\@GLSxtr@p@plural@
2730 \let\@glsxtrshort@\@glsxtr@p@short@
2731 \let\@Glsxtrshort@\@Glsxtr@p@short@
2732 \let\@GLSxtrshort@\@GLSxtr@p@short@
2733 \let\@glsxtrlong@\@glsxtr@p@long@
2734 \let\@Glsxtrlong@\@Glsxtr@p@long@
2735 \let\@GLSxtrlong@\@GLSxtr@p@long@
2736 \let\@glsxtrshortpl@\@glsxtr@p@shortpl@
2737 \let\@Glsxtrshortpl@\@Glsxtr@p@shortpl@
2738 \let\@GLSxtrshortpl@\@GLSxtr@p@shortpl@
2739 \let\@glsxtrlongpl@\@glsxtr@p@longpl@
2740 \let\@Glsxtrlongpl@\@Glsxtr@p@longpl@
2741 \let\@GLSxtrlongpl@\@GLSxtr@p@longpl@
2742 \let\@acrshort@\@glsxtr@p@acrshort@
2743 \let\@Acrshort@\@Glsxtr@p@acrshort@
2744 \let\@ACRshort@\@GLSxtr@p@acrshort@}
```

```

2745 \let\@acrshortpl\@glsxtr@p@acrshortpl@
2746 \let\@Acrshortpl\@Glsxtr@p@acrshortpl@
2747 \let\@ACRshortpl\@GLSxtr@p@acrshortpl@
2748 \let\@acrlong\@glsxtr@p@acrlong@
2749 \let\@Acrlong\@Glsxtr@p@acrlong@
2750 \let\@ACRlong\@GLSxtr@p@acrlong@
2751 \let\@acrlongpl\@glsxtr@p@acrlongpl@
2752 \let\@Acrlongpl\@Glsxtr@p@acrlongpl@
2753 \let\@ACRlongpl\@GLSxtr@p@acrlongpl@
2754 }

```

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

@glsxtr@p@text@

```
2755 \def\@glsxtr@p@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}
```

@Glsxtr@p@text@

```
2756 \def\@Glsxtr@p@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtr@p@text@

```
2757 \def\@GLSxtr@p@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtr@p@plural@

```
2758 \def\@lsxtr@p@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

lSxtr@p@plural@

```
2759 \def\@lSxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

LSxtr@p@plural@

```
2760 \def\@LSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glsxtr@p@short@

```

2761 \def\@glsxtr@p@short@#1#2[#3]{%
2762 {%
2763   \glssetabbrvfmt{\glscategory{#2}}%
2764   \glsabbrvfont{\gl Sentryshort{#2}}#3%
2765 }%
2766 }

```

Glsxtr@p@short@

```

2767 \def\@Glsxtr@p@short@#1#2[#3]{%
2768 {%
2769   \glssetabbrvfmt{\glscategory{#2}}%
2770   \glsabbrvfont{\Glsentryshort{#2}}#3%
2771 }%
2772 }

```

GLSxtr@p@short@

```
2773 \def\@GLSxtr@p@short@#1#2[#3]{%
2774   {%
2775     \glsetabbrvfmt{\glscategory{#2}}%
2776     \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshort{#2}}#3}%
2777   }%
2778 }
```

sxtr@p@shortpl@

```
2779 \def\@glsxtr@p@shortpl@#1#2[#3]{%
2780   {%
2781     \glsetabbrvfmt{\glscategory{#2}}%
2782     \glsabbrvfont{\glsentryshortpl{#2}}#3%
2783   }%
2784 }
```

Sxtr@p@shortpl@

```
2785 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2786   {%
2787     \glsetabbrvfmt{\glscategory{#2}}%
2788     \glsabbrvfont{\Glsentryshortpl{#2}}#3%
2789   }%
2790 }
```

Sxtr@p@shortpl@

```
2791 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2792   {%
2793     \glsetabbrvfmt{\glscategory{#2}}%
2794     \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshortpl{#2}}#3}%
2795   }%
2796 }
```

@glsxtr@p@long@

```
2797 \def\@glsxtr@p@long@#1#2[#3]{\{\glsentrylong{#2}#3}}
```

@Glsxtr@p@long@

```
2798 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@GLSxtr@p@long@

```
2799 \def\@GLSxtr@p@long@#1#2[#3]{%
2800   {\mfirstucMakeUppercase{\glslongfont{\glsentrylong{#2}}#3}}}
```

lsxtr@p@longpl@

```
2801 \def\@glsxtr@p@longpl@#1#2[#3]{\{\glsentrylongpl{#2}#3}}
```

lsxtr@p@longpl@

```
2802 \def\@Glsxtr@p@longpl@#1#2[#3]{\{\glslongfont{\Glsentrylongpl{#2}}#3}}
```

```

LSxtr@p@longpl@
2803 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2804   {\mfirstucMakeUppercase{\glslongfont{\glsentrylongpl{#2}}#3}}}

xtr@p@acrshort@
2805 \def\@glsxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\glsentryshort{#2}}#3}}

xtr@p@acrshort@
2806 \def\@GLSxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
2807 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2808   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
2809 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2810 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2811 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2812   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}

sxtr@p@acrlong@
2813 \def\@glsxtr@p@acrlong@#1#2[#3]{\{\glsentrylong{#2}}#3}}

sxtr@p@acrlong@
2814 \def\@Glsxtr@p@acrlong@#1#2[#3]{\{\Glsentrylong{#2}}#3}}

Sxtr@p@acrlong@
2815 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2816   {\mfirstucMakeUppercase{\glsentrylong{#2}}#3}}}

tr@p@acrlongpl@
2817 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\{\glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2818 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\{\Glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2819 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2820   {\mfirstucMakeUppercase{\glsentrylongpl{#2}}#3}}}

Commands to minimise conflict.

\@glsxtrp@opt
2821 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

```

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

```
2822 \newcommand*{\glsxtrsetpopts}[1]{%
2823   \renewcommand*{\@glsxtrp@opt}{#1}%
2824 }
```

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

```
2825 \newcommand*{\glossxtrsetpopts}{%
2826   \glsxtrsetpopts{noindex}%
2827 }
```

`\@glsxtrp`

```
2828 \newrobustcmd*{\@glsxtrp}[2]{%
```

Add scope.

```
2829   {%
2830     \let\glspostlinkhook\relax
2831     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
2832   }%
2833 }
```

`\@glsxtrp`

```
2834 \newrobustcmd*{\@glsxtrp}[2]{%
2835   \ifcsdef{gls#1}%
2836   {%
2837     \@glsxtrp{gls#1}{#2}%
2838   }%
2839   {%
2840     \ifcsdef{glsxtr#1}%
2841     {%
2842       \@glsxtrp{glsxtr#1}{#2}%
2843     }%
2844     {%
2845       \PackageError{glossaries-extra}{‘#1’ not recognised by
2846         \string\glsxtrp}{}%
2847     }%
2848   }%
2849 }
```

`\@Glsxtrp`

```
2850 \newrobustcmd*{\@Glsxtrp}[2]{%
2851   \ifcsdef{Gls#1}%
2852   {%
2853     \@glsxtrp{Gls#1}{#2}%
2854   }%
2855   {%
2856     \ifcsdef{Glsxtr#1}%
2857     {%
2858       \@glsxtrp{Glsxtr#1}{#2}%
2859     }%
2860   }%
2861 }
```



```

2860   {%
2861       \PackageError{glossaries-extra}{‘#1’ not recognised by
2862           \string\Glsxtrp}{}%
2863   }%
2864 }%
2865 }

```

\@GLSxtrp

```

2866 \newrobustcmd*{\@GLSxtrp}[2]{%
2867   \ifcsdef{GLS#1}%
2868   {%
2869       \@glsxtrp{GLS#1}{#2}%
2870   }%
2871   {%
2872       \ifcsdef{GLSxtr#1}%
2873       {%
2874           \@glsxtrp{GLSxtr#1}{#2}%
2875       }%
2876       {%
2877           \PackageError{glossaries-extra}{‘#1’ not recognised by
2878               \string\GLSxtrp}{}%
2879       }%
2880   }%
2881 }

```

\glsxtr@entry@p

```

2882 \newrobustcmd*{\glsxtr@headentry@p}[2]{%
2883   \glsifattribute{#1}{headuc}{true}%
2884   {%
2885       \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
2886   }%
2887   {%
2888       \@gls@entry@field{#1}{#2}%
2889   }%
2890 }

```

\glsxtrp Not robust as it needs to expand somewhat.

```

2891 \ifdef\texorpdfstring
2892 {
2893   \newcommand{\glsxtrp}[2]{%
2894     \protect\NoCaseChange
2895     {%
2896         \protect\texorpdfstring
2897         {%
2898             \protect\glsxtrifinmark
2899             {%
2900                 \ifcsdef{glsxtrhead#1}%
2901                 {%
2902                     {\protect\csuse{glsxtrhead#1}{#2}}%

```

```

2903         }%
2904         {%
2905             \glxtr@headentry@p{#2}{#1}%
2906         }%
2907     }%
2908     {%
2909         \@glxtrp{#1}{#2}%
2910     }%
2911 }%
2912 {%
2913     \protect\@gls@entry@field{#2}{#1}%
2914 }%
2915 }%
2916 }
2917 }
2918 {
2919     \newcommand{\glxtrp}[2]{%
2920         \protect\NoCaseChange
2921         {%
2922             \protect\glxtrifinmark
2923             {%
2924                 \ifcsdef{glxtrhead#1}%
2925                 {%
2926                     {\protect\csuse{glxtrhead#1}}%
2927                 }%
2928                 {%
2929                     \glxtr@headentry@p{#2}{#1}%
2930                 }%
2931             }%
2932             {%
2933                 \@glxtrp{#1}{#2}%
2934             }%
2935         }%
2936     }
2937 }

```

Provide short synonyms for the most common option.

`\glsp`

```
2938 \newcommand*{\glsp}{\glxtrp{short}}
```

`\glsp`

```
2939 \newcommand*{\glsp}{\glxtrp{text}}
```

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

```
2940 \ifdef\texorpdfstring
```

```
2941 {
```

```
2942     \newcommand{\Glsxtrp}[2]{%
```

```

2943 \protect\NoCaseChange
2944 {%
2945 \protect\texorpdfstring
2946 {%
2947 \protect\glstrifinmark
2948 {%
2949 \ifcsdef{Glsxtrhead#1}%
2950 {%
2951 {\protect\csuse{Glsxtrhead#1}{#2}}%
2952 }%
2953 {%
2954 \protect\@Gls@entry@field{#2}{#1}%
2955 }%
2956 }%
2957 {%
2958 \@Glsxtrp{#1}{#2}%
2959 }%
2960 }%
2961 {%
2962 \protect\@gls@entry@field{#2}{#1}%
2963 }%
2964 }%
2965 }
2966 }
2967 {
2968 \newcommand{\Glsxtrp}[2]{%
2969 \protect\NoCaseChange
2970 {%
2971 \protect\glstrifinmark
2972 {%
2973 \ifcsdef{Glsxtrhead#1}%
2974 {%
2975 {\protect\csuse{Glsxtrhead#1}}%
2976 }%
2977 {%
2978 \protect\@Gls@entry@field{#2}{#1}%
2979 }%
2980 }%
2981 {%
2982 \@Glsxtrp{#1}{#2}%
2983 }%
2984 }%
2985 }
2986 }

```

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

```

2987 \ifdef\texorpdfstring
2988 {
2989 \newcommand{\Glsxtrp}[2]{%

```

```

2990 \protect\NoCaseChange
2991 {%
2992 \protect\texorpdfstring
2993 {%
2994 \protect\glsxtrifinmark
2995 {%
2996 \ifcsdef{GLSxtr#1}%
2997 {%
2998 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2999 }%
3000 {%
3001 \protect\mfirstucMakeUppercase
3002 {%
3003 \protect\@gls@entry@field{#2}{#1}%
3004 }%
3005 }%
3006 }%
3007 {%
3008 \@GLSxtrp{#1}{#2}%
3009 }%
3010 }%
3011 {%
3012 \protect\@gls@entry@field{#2}{#1}%
3013 }%
3014 }%
3015 }
3016 }
3017 {
3018 \newcommand{\GLSxtrp}[2]{%
3019 \protect\NoCaseChange
3020 {%
3021 \protect\glsxtrifinmark
3022 {%
3023 \ifcsdef{GLSxtr#1}%
3024 {%
3025 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
3026 }%
3027 {%
3028 \protect\mfirstucMakeUppercase
3029 {%
3030 \protect\@gls@entry@field{#2}{#1}%
3031 }%
3032 }%
3033 }%
3034 {%
3035 \@GLSxtrp{#1}{#2}%
3036 }%
3037 }%
3038 }

```

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, but changing the flag can cause problems in certain situations, so to allow the normal unsetting to be temporarily disabled, \@glsunset is let to \@glsxtr@unset, which performs the actual unsetting through \@@glsunset and then does the hook. This means that the unsetting (and the hook) can switched off by redefining \@glsunset and then switched back on again by changing the definition back to \@glsxtr@unset.

```
\@glsxtr@unset  Global unset.
3040 \newcommand*{\@glsxtr@unset}[1]{%
3041   \@@glsunset{#1}%
3042   \glsxtrpostunset{#1}%
3043 }%

\@glsunset  Global unset.
3044 \let\@glsunset\@glsxtr@unset

glsxtrpostunset
3045 \newcommand*{\glsxtrpostunset}[1]{%

    Provide a command to store a list of labels that will need unsetting.

tUnsetBuffering
3046 \newcommand*{\GlsXtrStartUnsetBuffering}{%
3047   \let\@glsxtr@org@unset@buffer\@glsxtr@unset@buffer
3048   \def\@glsxtr@unset@buffer{}%
3049   \let\@glsunset\@glsxtrbuffer@unset
3050 }

xtrbuffer@unset  This must use a global change since \gls may have to be placed inside \mbox (for example,
                  with soul commands).
3051 \newcommand*{\@glsxtrbuffer@unset}[1]{%
3052   \listxadd\@glsxtr@unset@buffer{#1}%
3053 }

pUnsetBuffering
3054 \newcommand*{\GlsXtrStopUnsetBuffering}{%
3055   \@ifstar\s@GlsXtrStopUnsetBuffering\@GlsXtrStopUnsetBuffering
3056 }
```

pUnsetBuffering Unstarred form (global unset).

```
3057 \newcommand*{\@GlsXtrStopUnsetBuffering}{%
3058   \let\@glsunset\@glxtr@unset
3059   \forlistloop\@glsunset\@glxtr@unset@buffer
3060   \let\@glxtr@unset@buffer\@glxtr@org@unset@buffer
3061 }
```

pUnsetBuffering Starred form (local unset).

```
3062 \newcommand*{\s@GlsXtrStopUnsetBuffering}{%
3063   \forlistloop\@glsllocalunset\@glxtr@unset@buffer
3064   \let\@glsunset\@glxtr@unset
3065 }
```

\@glsllocalunset Local unset.

```
3066 \renewcommand*{\@glsllocalunset}[1]{%
3067   \@@glsllocalunset{#1}%
3068   \glxtrpostlocalunset{#1}%
3069 }%
```

rpostlocalunset

```
3070 \newcommand*{\glxtrpostlocalunset}[1]{}
```

\@glsrreset Global reset.

```
3071 \renewcommand*{\@glsrreset}[1]{%
3072   \@@glsrreset{#1}%
3073   \glxtrpostreset{#1}%
3074 }%
```

glxtrpostreset

```
3075 \newcommand*{\glxtrpostreset}[1]{}
```

\@glsllocalreset Local reset.

```
3076 \renewcommand*{\@glsllocalreset}[1]{%
3077   \@@glsllocalreset{#1}%
3078   \glxtrpostlocalreset{#1}%
3079 }%
```

rpostlocalreset

```
3080 \newcommand*{\glxtrpostlocalreset}[1]{}
```

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

```
3081 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

 Enable entry counting:

```
3082   \glsenableentrycount
```

Redefine \gls etc:

```
3083 \renewcommand*{\gls}{\cglsl}%
3084 \renewcommand*{\Gls}{\cGls}%
3085 \renewcommand*{\glspl}{\cglspl}%
3086 \renewcommand*{\Glspl}{\cGlspl}%
3087 \renewcommand*{\GLS}{\cGLS}%
3088 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
3089 \@glsxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
3090 \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
3091 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
3092 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
3093 can't be used with \string\GlsXtrEnableEntryCounting}%
3094 {Use one or other but not both commands}}%
3095 }
```

ycountunsetattr

```
3096 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
3097 \@for\@glsxtr@cat:=#1\do
3098 {%
3099 \ifdefempty{\@glsxtr@cat}{}%
3100 {%
3101 \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3102 }%
3103 }%
3104 }
```

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

nableentrycount

```
3105 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
3106 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
```

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

```
3107 \renewcommand*{\gls@defdocnewglossaryentry}{%
3108 \renewcommand*\newglossaryentry[2]{%
3109 \PackageError{glossaries}{\string\newglossaryentry\space
3110 may only be used in the preamble when entry counting has
3111 been activated}{If you use \string\glsenableentrycount\space
3112 you must place all entry definitions in the preamble not in
3113 the document environment}%
3114 }%
3115 }
```

New commands to access new fields:

```
3116 \newcommand*{\glsentrycurrcount}[1]{%
```

```

3117 \ifcsundef{glo@glsdetoklabel{##1}@currcount}%
3118 {0}{\@gls@entry@field{##1}{currcount}}}%
3119 }%
3120 \newcommand*{\glsentryprevcount}[1]{%
3121 \ifcsundef{glo@glsdetoklabel{##1}@prevcount}%
3122 {0}{\@gls@entry@field{##1}{prevcount}}}%
3123 }%

```

Adjust post unset and reset:

```

3124 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
3125 \renewcommand*{\glsxtrpostunset}[1]{%
3126 \@glsxtr@entrycount@org@unset{##1}%
3127 \@gls@increment@currcount{##1}%
3128 }%
3129 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
3130 \renewcommand*{\glsxtrpostlocalunset}[1]{%
3131 \@glsxtr@entrycount@org@localunset{##1}%
3132 \@gls@local@increment@currcount{##1}%
3133 }%
3134 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
3135 \renewcommand*{\glsxtrpostreset}[1]{%
3136 \@glsxtr@entrycount@org@reset{##1}%
3137 \csgdef{glo@glsdetoklabel{##1}@currcount}{0}%
3138 }%
3139 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
3140 \renewcommand*{\glsxtrpostlocalreset}[1]{%
3141 \@glsxtr@entrycount@org@localreset{##1}%
3142 \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
3143 }%

```

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

```

3144 \let\@cgl@s\@cgl@s
3145 \let\@cgl spl\@cgl spl
3146 \let\@cGls\@cGls
3147 \let\@cGlspl\@cGlspl
3148 \let\@cGLS\@cGLS
3149 \let\@cGLSpl\@cGLSpl

```

The rest is as the original definition.

```

3150 \AtEndDocument{\@gls@write@entrycounts}%
3151 \renewcommand*{\@gls@entry@count}[2]{%
3152 \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
3153 }%
3154 \let\glsenableentrycount\relax
3155 \renewcommand*{\glsenableentryunitcount}{%
3156 \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space
3157 can't be used with \string\glsenableentrycount}%
3158 {Use one or other but not both commands}%
3159 }%

```


3160 }

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.

```
3161 \renewcommand*{\@gls@write@entrycounts}{%
3162   \immediate\write\@auxout
3163     {\string\providecommand*{\string\@gls@entry@count}[2]{}}%
3164   \count@=0\relax
3165   \forallglsentries{\@glsentry}{%
3166     \glshasattribute{\@glsentry}{entrycount}%
3167     {%
3168       \ifglsused{\@glsentry}%
3169       {%
3170         \immediate\write\@auxout
3171           {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
3172       }%
3173     }%
3174     \advance\count@ by \@ne
3175   }%
3176 }%
3177 }%
3178 \ifnum\count@=0
3179   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3180   \MessageBreak with \string\glsenableentrycount\space but the
3181   \MessageBreak attribute 'entrycount' hasn't
3182   \MessageBreak been assigned to any of the defined
3183   \MessageBreak entries}%
3184 \fi
3185 }
```

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

```
3186 \newcommand*{\glxtrifcounttrigger}[3]{%
3187   \glshasattribute{#1}{entrycount}%
3188   {%
3189     \ifnum\glsentryprevcount{#1}>\glsgetattribute{#1}{entrycount}\relax
3190     #3%
3191   \else
3192     #2%
3193   \fi
3194 }%
3195 {#3}%
3196 }
```

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

\@@cgl@s@

```
3197 \def\@@cgl@s@#1#2[#3]{%
3198   \glxtrifcounttrigger{#2}%
3199   {%
3200     \cgl@sformat{#2}{#3}%
3201     \gl sunset{#2}%
3202   }%
3203   {%
3204     \@gls@{#1}{#2}[#3]%
3205   }%
3206 }%
```

\@@cgl spl@

```
3207 \def\@@cgl spl@#1#2[#3]{%
3208   \glxtrifcounttrigger{#2}%
3209   {%
3210     \cgl spl format{#2}{#3}%
3211     \gl sunset{#2}%
3212   }%
3213   {%
3214     \@glspl@{#1}{#2}[#3]%
3215   }%
3216 }%
```

\@@cGls@

```
3217 \def\@@cGls@#1#2[#3]{%
3218   \glxtrifcounttrigger{#2}%
3219   {%
3220     \cGlsformat{#2}{#3}%
3221     \gl sunset{#2}%
3222   }%
3223   {%
3224     \@Gls@{#1}{#2}[#3]%
3225   }%
3226 }%
```

\@@cGlspl@

```
3227 \def\@@cGlspl@#1#2[#3]{%
3228   \glxtrifcounttrigger{#2}%
3229   {%
3230     \cGlsplformat{#2}{#3}%
3231     \gl sunset{#2}%
3232   }%
3233   {%
3234     \@Glspl@{#1}{#2}[#3]%
3235   }%
3236 }%
```

\@@cGLS@

```

3237 \def\@@cGLS@#1#2[#3]{%
3238   \glstrifcounttrigger{#2}%
3239   {%
3240     \cGLSformat{#2}{#3}%
3241     \glset{#2}%
3242   }%
3243   {%
3244     \@GLS@{#1}{#2}[#3]%
3245   }%
3246 }%

```

\@@cGLSpl@

```

3247 \def\@@cGLSpl@#1#2[#3]{%
3248   \glstrifcounttrigger{#2}%
3249   {%
3250     \cGLSplformat{#2}{#3}%
3251     \glset{#2}%
3252   }%
3253   {%
3254     \@GLSpl@{#1}{#2}[#3]%
3255   }%
3256 }%

```

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

\@cgl's@

```

3257 \def\@cgl's@#1#2[#3]{\@gl's@{#1}{#2}[#3]}

```

\@cGls@

```

3258 \def\@cGls@#1#2[#3]{\@Gls@{#1}{#2}[#3]}

```

\@cgl'spl@

```

3259 \def\@cgl'spl@#1#2[#3]{\@gl'spl@{#1}{#2}[#3]}

```

\@cGlspl@

```

3260 \def\@cGlspl@#1#2[#3]{\@Glspl@{#1}{#2}[#3]}

```

Add all upper case versions not provided by glossaries.

\cGLS

```

3261 \newrobustcmd*{\cGLS}{\@gl's@hyp@opt\@cGLS}

```

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

```

3262 \newcommand*{\@cGLS}[2][ ]{%
3263   \new@ifnextchar[\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2}[ ]}%
3264 }

```

```

\@cGLS@
3265 \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.
3266 \newcommand*\cGLSformat[2]{%
3267   \expandafter\mfirstuc\MakeUppercase\expandafter{\cGLSformat{#1}{#2}}}%
3268 }

\cGLSpl
3269 \newrobustcmd*\cGLSpl{\@gls@hyp@opt\@cGLSpl}

\@cGLSpl  Defined the un-starred form. Need to determine if there is a final optional argument
3270 \newcommand*\@cGLSpl[2][{}]{%
3271   \new@ifnextchar[\@cGLSpl@{#1}{#2}}{\@cGLSpl@{#1}{#2}[{}]}%
3272 }

\@cGLSpl@
3273 \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.
3274 \newcommand*\cGLSplformat[2]{%
3275   \expandafter\mfirstuc\MakeUppercase\expandafter{\cGLSplformat{#1}{#2}}}%
3276 }

      Modify the trigger formats to check for the regular attribute.

\cGLSformat
3277 \renewcommand*\cGLSformat[2]{%
3278   \glsifregular{#1}
3279   {\glsentryfirst{#1}}%
3280   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
3281 }

\cGlsformat
3282 \renewcommand*\cGlsformat[2]{%
3283   \glsifregular{#1}
3284   {\Glsentryfirst{#1}}%
3285   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
3286 }

\cGLSplformat
3287 \renewcommand*\cGLSplformat[2]{%
3288   \glsifregular{#1}
3289   {\glsentryfirstplural{#1}}%
3290   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
3291 }

```

\cGlsplformat

```
3292 \renewcommand*{\cGlsplformat}[2]{%
3293   \glsifregular{#1}
3294   {\Glsentryfirstplural{#1}}%
3295   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2}%
3296 }
```

New code similar to above for unit counting.

defunitcounters

```
3297 \newcommand*{\@@newglossaryentry@defunitcounters}{%
3298   \edef\@glo@countunit{\csuse{@glxtr@categoryattr@@\@glo@category @unitcount}}%
3299   \ifdefined\@glo@countunit
3300     {}%
3301     {%
3302       \@glxtr@ifunitcounter{\@glo@countunit}%
3303       {}%
3304       {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}%
3305     }%
3306 }
```

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

```
3307 \newcommand*{\@glxtr@unitcountlist}{}%
```

@addunitcounter

```
3308 \newcommand*{\@glxtr@addunitcounter}[1]{%
3309   \listadd{\@glxtr@unitcountlist}{#1}%
3310   \ifcsundef{glxtr@theunit@#1}
3311     {%
3312       \ifcsdef{theH#1}%
3313       {\csdef{glxtr@theunit@#1}{\csuse{theH#1}}}%
3314       {\csdef{glxtr@theunit@#1}{\csuse{the#1}}}%
3315     }%
3316   {}%
3317 }
```

r@ifunitcounter

```
3318 \newcommand*{\@glxtr@ifunitcounter}[3]{%
3319   \xifinlist{#1}{\@glxtr@unitcountlist}{#2}{#3}%
3320 }
```

urrentunitcount

```
3321 \newcommand*{\@glxtr@currentunitcount}[1]{%
3322   glo@\glsetoklabel{#1}@currunit@\glsetattribute{#1}{unitcount}.%
3323   \csuse{glxtr@theunit@\glsetattribute{#1}{unitcount}}%
3324 }
```

previousunitcount

```
3325 \newcommand*\@glxtr@previousunitcount[1]{%
3326   glo@\glstdetoklabel{#1}@prevunit@\glsggetattribute{#1}{unitcount}.%
3327   \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}}%
3328 }
```

t@currunitcount

```
3329 \newcommand*\@gl@increment@currunitcount[1]{%
3330   \glshasattribute{#1}{unitcount}%
3331   {%
3332     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
3333     \ifcsundef{\@glxtr@csname}%
3334     {%
3335       \csgdef{\@glxtr@csname}{1}%
3336       \listcsxadd
3337         {glo@\glstdetoklabel{#1}@unitlist}%
3338         {\glsggetattribute{#1}{unitcount}.%
3339         \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}}%
3340       }%
3341     }%
3342     {%
3343       \csxdef{\@glxtr@csname}%
3344       {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
3345     }%
3346   }%
3347   {}%
3348 }
```

t@currunitcount

```
3349 \newcommand*\@gl@local@increment@currunitcount[1]{%
3350   \glshasattribute{#1}{unitcount}%
3351   {%
3352     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
3353     \ifcsundef{\@glxtr@csname}%
3354     {%
3355       \csdef{\@glxtr@csname}{1}%
3356       \listcseadd
3357         {glo@\glstdetoklabel{#1}@unitlist}%
3358         {\glsggetattribute{#1}{unitcount}.%
3359         \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}}%
3360       }%
3361     }%
3362     {%
3363       \csedef{\@glxtr@csname}%
3364       {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
3365     }%
3366   }%
3367   {}%
3368 }
```

r@currunitcount

```
3369 \newcommand*{\@glxstr@currunitcount}[2]{%
3370   \ifcsundef
3371   {glo@\glstetoklabel{#1}@currunit@#2}%
3372   {0}%
3373   {\csuse{glo@\glstetoklabel{#1}@currunit@#2}}%
3374 }%
```

r@prevunitcount

```
3375 \newcommand*{\@glxstr@prevunitcount}[2]{%
3376   \ifcsundef
3377   {glo@\glstetoklabel{#1}@prevunit@#2}%
3378   {0}%
3379   {\csuse{glo@\glstetoklabel{#1}@prevunit@#2}}%
3380 }%
```

entryunitcount

```
3381 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
3382   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
3383   \renewcommand*{\glstetoknewglossaryentry}{%
3384     \renewcommand*{\newglossaryentry}[2]{%
3385       \PackageError{glossaries}{\string\newglossaryentry\space
3386         may only be used in the preamble when entry counting has
3387         been activated}{If you use \string\glsenableentryunitcount\space
3388         you must place all entry definitions in the preamble not in
3389         the document environment}%
3390     }%
3391   }%
```

New commands to access new fields:

```
3392   \newcommand*{\glstentrycurrcount}[1]{%
3393     \@glxstr@currunitcount{##1}{\glstgetattribute{##1}{unitcount}}.%
3394     \csuse{\glxstr@theunit@\glstgetattribute{##1}{unitcount}}}%
3395   }%
3396   \newcommand*{\glstentryprevcount}[1]{%
3397     \@glxstr@prevunitcount{##1}{\glstgetattribute{##1}{unitcount}}.%
3398     \csuse{\glxstr@theunit@\glstgetattribute{##1}{unitcount}}}%
3399   }%
```

Access total count:

```
3400   \newcommand*{\glstentryprevtotalcount}[1]{%
3401     \ifcsundef{glo@\glstetoklabel{##1}@prevunittotal}%
3402     {0}%
3403     {%
3404       \number\csuse{glo@\glstetoklabel{##1}@prevunittotal}
3405     }%
3406   }%
```

Access max value:

```
3407 \newcommand*{\glstentryprevmaxcount}[1]{%
3408   \ifcsundef{glo@\glstetoklabel{##1}@prevunitmax}%
3409     {0}%
3410     {%
3411       \number\csuse{glo@\glstetoklabel{##1}@prevunitmax}
3412     }%
3413   }%
```

Adjust post unset and reset:

```
3414 \let\@glstxtr@entryunitcount@org@unset\glstxtrpostunset
3415 \renewcommand*{\glstxtrpostunset}[1]{%
3416   \@glstxtr@entryunitcount@org@unset{##1}%
3417   \@glst@increment@currunitcount{##1}%
3418 }%
3419 \let\@glstxtr@entryunitcount@org@localunset\glstxtrpostlocalunset
3420 \renewcommand*{\glstxtrpostlocalunset}[1]{%
3421   \@glstxtr@entryunitcount@org@localunset{##1}%
3422   \@glst@local@increment@currunitcount{##1}%
3423 }%
3424 \let\@glstxtr@entryunitcount@org@reset\glstxtrpostreset
3425 \renewcommand*{\glstxtrpostreset}[1]{%
3426   \glshasattribute{##1}{unitcount}%
3427   {%
3428     \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
3429     \ifcsundef{\@glstxtr@csname}%
3430       {}%
3431       {\csgdef{\@glstxtr@csname}{0}}%
3432     }%
3433   {}%
3434 }%
3435 \let\@glstxtr@entryunitcount@org@localreset\glstxtrpostlocalreset
3436 \renewcommand*{\glstxtrpostlocalreset}[1]{%
3437   \@glstxtr@entryunitcount@org@localreset{##1}%
3438   \glshasattribute{##1}{unitcount}%
3439   {%
3440     \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
3441     \ifcsundef{\@glstxtr@csname}%
3442       {}%
3443       {\csdef{\@glstxtr@csname}{0}}%
3444     }%
3445   {}%
3446 }%
```

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

```
3447 \let\@cglst@\@cglst@
3448 \let\@cglstpl@\@cglstpl@
3449 \let\@cGls@\@cGls@
```



```

3450 \let\@cGlspl@\@cGlspl@
3451 \let\@cGLS@\@cGLS@
3452 \let\@cGLSpl@\@cGLSpl@

```

Write information to the aux file.

```

3453 \AtEndDocument{\@gls@write@entryunitcounts}%
3454 \renewcommand*{\@gls@entry@unitcount}[3]{%
3455   \csgdef{glo@\glsdetoklabel{##1}@prevunit@##3}{##2}%
3456   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
3457   {\csgdef{glo@\glsdetoklabel{##1}@prevunittotal}{##2}}%
3458   {%
3459     \csxdef{glo@\glsdetoklabel{##1}@prevunittotal}{
3460       \number\numexpr\csuse{glo@\glsdetoklabel{##1}@prevunittotal}+##2}%
3461     }%
3462     \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
3463     {\csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}}%
3464     {%
3465       \ifnum\csuse{glo@\glsdetoklabel{##1}@prevunitmax}<##2
3466       \csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}%
3467     }%
3468   }%
3469 }%
3470 \let\glsenableentryunitcount\relax
3471 \renewcommand*{\glsenableentrycount}{%
3472   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
3473     can't be used with \string\glsenableentryunitcount}%
3474   {Use one or other but not both commands}%
3475 }%
3476 }
3477 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

3478 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

3479 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
3480   \immediate\write\@auxout
3481   {\string\@gls@entry@unitcount
3482     {\@glsentry}%
3483     {\@glsxtr@currunitcount{\@glsentry}{#1}%
3484     }%
3485     {#1}}%
3486 }

```

entryunitcounts

```

3487 \newcommand*{\@gls@write@entryunitcounts}{%
3488   \immediate\write\@auxout
3489   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
3490   \count@=0\relax

```

```

3491 \forallglentries{\@glentry}{%
3492   \glshasattribute{\@glentry}{unitcount}%
3493   {%
3494     \ifglused{\@glentry}%
3495     {%
3496       \forlistcsloop
3497         {\@gls@write@entryunitcounts@do}%
3498         {glo@\glsetoklabel{\@glentry}@unitlist}%
3499     }%
3500   }%
3501   \advance\count@ by \@ne
3502 }%
3503 {}%
3504 }%
3505 \ifnum\count@=0
3506   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3507   \MessageBreak with \string\glsenableentryunitcount\space but the
3508   \MessageBreak attribute ‘unitcount’ hasn’t
3509   \MessageBreak been assigned to any of the defined
3510   \MessageBreak entries}%
3511 \fi
3512 }

```

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.

```

3513 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

3514 \glsenableentryunitcount

```

Redefine \gls etc:

```

3515 \renewcommand*{\gls}{\cgl}%
3516 \renewcommand*{\Gls}{\cGls}%
3517 \renewcommand*{\glspl}{\cglspl}%
3518 \renewcommand*{\Glspl}{\cGlspl}%
3519 \renewcommand*{\GLS}{\cGLS}%
3520 \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

3521 \@glxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

3522 \let\GlsXtrEnableEntryUnitCounting\@glxtr@setentryunitcountunsetattr
3523 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
3524   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
3525     can’t be used with \string\GlsXtrEnableEntryUnitCounting}%
3526   {Use one or other but not both commands}}%
3527 }

```

tcountunsetattr

```

3528 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
3529   \@for\@glsxtr@cat:=#1\do
3530   {%
3531     \ifdefempty{\@glsxtr@cat}{}%
3532     {%
3533       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3534       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
3535     }%
3536   }%
3537 }

```

1.3.6 Acronym Modifications

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

`\SetGenericNewAcronym`

```

3538 \renewcommand*{\SetGenericNewAcronym}{%
3539   \let\@Gls@entryname\@Gls@acrentryname
3540   \renewcommand{\newacronym}[4][{}]{%
3541     \ifdefempty{\@glsacronymlists}%
3542     {%
3543       \def\@glo@type{\acronymtype}%
3544       \setkeys{glossentry}{##1}%
3545       \DeclareAcronymList{\@glo@type}%
3546     }%
3547   }%
3548   \glskeylisttok{##1}%
3549   \glslabeltok{##2}%
3550   \glsshorttok{##3}%
3551   \glslongtok{##4}%
3552   \newacronymhook
3553   \protected@edef\@do@newglossaryentry{%
3554     \noexpand\newglossaryentry{\the\glslabeltok}%
3555     {%
3556       type=\acronymtype,%
3557       name={\expandonce{\acronymentry{##2}}},%
3558       sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
3559       text={\the\glsshorttok},%
3560       short={\the\glsshorttok},%
3561       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3562       long={\the\glslongtok},%
3563       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3564       category=acronym,

```

```

3565     \GenericAcronymFields,%
3566     \the\glskeylisttok
3567 }%
3568 }%
3569 \do@newglossaryentry
3570 }%
3571 \renewcommand*{\acrfullfmt}[3]{%
3572   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
3573 \renewcommand*{\Acrfullfmt}[3]{%
3574   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
3575 \renewcommand*{\ACRfullfmt}[3]{%
3576   \glslink[##1]{##2}{%
3577     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
3578 \renewcommand*{\acrfullplfmt}[3]{%
3579   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
3580 \renewcommand*{\Acrfullplfmt}[3]{%
3581   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
3582 \renewcommand*{\ACRfullplfmt}[3]{%
3583   \glslink[##1]{##2}{%
3584     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
3585 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}}%
3586 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
3587 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}}%
3588 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
3589 }

```

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:

```

3590 \let\@glstr@org@setacronymstyle\setacronymstyle
3591 \let\@glstr@org@newacronymstyle\newacronymstyle

```

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

```

3592 \newcommand*{\MakeAcronymsAbbreviations}{%
3593   \renewcommand*{\newacronym}[4][{}]{%
3594     \glstr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
3595   }%
3596   \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
3597   \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%
3598   \renewcommand*{\setacronymstyle}[1]{%
3599     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}
3600     unavailable.
3601     Use \string\setabbreviationstyle\space instead.
3602     The original acronym interface can be restored with
3603     \string\RestoreAcronyms}{}%
3604   }%
3605   \renewcommand*{\newacronymstyle}[1]{%

```

```

3606      \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
3607      available unless you restore the original acronym interface with
3608      \string\RestoreAcronyms}%
3609      \@glstr@org@newacronymstyle{##1}%
3610    }%
3611 }

```

Switch acronyms to abbreviations:

```

3612 \MakeAcronymsAbbreviations

```

RestoreAcronyms Restore acronyms to glossaries interface.

```

3613 \newcommand*{\RestoreAcronyms}{%
3614   \SetGenericNewAcronym
3615   \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
3616   \renewcommand{\acronymfont}[1]{##1}%
3617   \let\setacronymstyle\@glstr@org@setacronymstyle
3618   \let\newacronymstyle\@glstr@org@newacronymstyle

```

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

```

3619   \renewcommand*\@gls@link@checkfirsthyper{%
3620     \ifglused{\glslabel}%
3621     {\let\glstrifwasfirstuse\@secondoftwo}
3622     {\let\glstrifwasfirstuse\@firstoftwo}%
3623     \@glstr@org@checkfirsthyper
3624   }
3625   \glssetcategoryattribute{acronym}{regular}{false}%
3626   \setacronymstyle{long-short}%
3627 }

```

\glsacspace Allow the user to customise the maximum value.

```

3628 \renewcommand*{\glsacspace}[1]{%
3629   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
3630   \ifdim\dimen@<\glsacspacemax~\else\space\fi
3631 }

```

\glsacspacemax Value used in the above.

```

3632 \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

```

3633 \newcommand*{\@glstr@reg@glosslist}{}

```

Save the original definition of `\makeglossaries`:

```
3634 \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
```

```
3635 \renewcommand*{\makeglossaries}[1] [] {%
3636   \ifx\@glxtr@record@setting\@glxtr@record@setting@only
3637   \PackageError{glossaries-extra}{\string\makeglossaries\space
3638     not permitted\MessageBreak with record=only package option}%
3639   {You may only use \string\makeglossaries\space with
3640     record=off or record=alsoindex options}%
3641   \else
3642     \ifblank{#1}%
3643     {\@glxtr@org@makeglossaries}%
3644     {%
3645       \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
3646       \PackageError{glossaries-extra}{\string\makeglossaries[#1]\space
3647         not permitted\MessageBreak with record=alsoindex package option}%
3648       {You may only use the hybrid \string\makeglossaries[...]\space with
3649         record=off option}%
3650     \else
3651       \edef\@glxtr@reg@glosslist{#1}%
3652       \ifundef{\glswrite}{\newwrite\glswrite}{}%
3653       \protected@write\@auxout{}{\string\providecommand
3654         \string\@glxtr@order[1]{}%
3655       \protected@write\@auxout{}{\string\providecommand
3656         \string\@istfilename[1]{}%
3657       \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
3658       \protected@write\@auxout{}{\string\@glxtr@order{\glxtr@order}}%
3659       \protected@write\@auxout{}{\string\glxtr@makeglossaries{#1}}%
3660       \write\@auxout{\string\providecommand\string\@glxtr@reference[3]{}%}
```

Iterate through each supplied glossary type and activate it.

```
3661   \@for\@glo@type:=#1\do{%
3662     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
3663   }%
```

New glossaries must be created before `\makeglossaries`:

```
3664   \renewcommand*\newglossary[4] [] {%
3665   \PackageError{glossaries}{New glossaries
3666     must be created before \string\makeglossaries}{You need
3667     to move \string\makeglossaries\space after all your
3668     \string\newglossary\space commands}}%
```

Any subsequence instances of this command should have no effect

```
3669   \let\@makeglossary\relax
3670   \let\makeglossary\relax
```

```

3671 \renewcommand\makeglossaries[1][]{%
  Disable all commands that have no effect after \makeglossaries
3672 \@disable@onlypremakeg
  Allow see key:
3673 \let\gls@checkseeallowed\relax
  Adjust \@do@seeglossary. This needs to check for the entry's existence but don't increment
  associated counter.
3674 \renewcommand*\@do@seeglossary}[2]{%
3675 \glsdoifexists{##1}%
3676 {%
3677 \edef\@gls@label{\glsdetoklabel{##1}}%
3678 \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3679 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3680 {\@glsxtr@org@doseeglossary{##1}{##2}}%
3681 {%
3682 \@@glsxtrwrglossmark
3683 \protected@write\@auxout{}{%
3684 \string\@gls@reference
3685 {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}}%
3686 }%
3687 }%
3688 }%
3689 }%
  Adjust \@do@@wrglossary
3690 \let\@glsxtr@@do@@wrglossary\@do@@wrglossary
3691 \def\@do@@wrglossary{%
3692 \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3693 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3694 {\@glsxtr@@do@@wrglossary}%
3695 {\gls@noidxglossary}%
3696 }%
  Suppress warning about no \makeglossaries
3697 \let\warn@nomakeglossaries\relax
3698 \def\warn@noprintglossary{%
3699 \GlossariesWarningNoLine{No \string\printglossary\space
3700 or \string\printglossaries\space
3701 found.^^J(Remove \string\makeglossaries\space if you don't want
3702 any glossaries.)^^JThis document will not have a glossary}%
3703 }%
  Only warn for glossaries not listed.
3704 \renewcommand{\@gls@noref@warn}[1]{%
3705 \edef\@gls@type{##1}%
3706 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3707 {%
3708 \GlossariesExtraWarning{Can't use
3709 \string\printnoidxglossary[type={\@gls@type}]

```

```

3710         when '@gls@type' is listed in the optional argument of
3711         \string\makeglossaries}%
3712     }%
3713     {%
3714         \GlossariesWarning{Empty glossary for
3715         \string\printnoidxglossary[type={##1}].
3716         Rerun may be required (or you may have forgotten to use
3717         commands like \string\gls)}%
3718     }%
3719 }%

```

Adjust display number list to check for type:

```

3720 \renewcommand*{\glsdisplaynumberlist}[1]{%
3721 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3722 {\@glsxtr@idx@displaynumberlist{##1}}%
3723 {\@glsxtr@noidx@displaynumberlist{##1}}%
3724 }%

```

Adjust entry list:

```

3725 \renewcommand*{\glsentrynumberlist}[1]{%
3726 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3727 {\@glsxtr@idx@entrynumberlist{##1}}%
3728 {\@glsxtr@noidx@entrynumberlist{##1}}%
3729 }%

```

Adjust number list loop

```

3730 \renewcommand*{\glsnumberlistloop}[2]{%
3731 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3732 {%
3733     \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3734     not available for glossary '@##1'}{%
3735 }%
3736 {\@glsxtr@noidx@numberlistloop{##1}{##2}}%
3737 }%

```

Only sanitize sort for normal indexing glossaries.

```

3738 \renewcommand*{\glsprestandardsort}[3]{%
3739 \expandafter\DTLifinlist\expandafter{##2}{\@glsxtr@reg@glosslist}%
3740 {%
3741     \glsdosanitizesort
3742 }%
3743 {%
3744     \ifglssanitizesort
3745     \@gls@noidx@sanitizesort
3746     \else
3747     \@gls@noidx@nosanitizesort
3748     \fi
3749 }%
3750 }%

```

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


```

3751 \renewcommand*\new@glossaryentry[2]{%
3752 \PackageError{glossaries-extra}{Glossary entries must be defined
3753 in the preamble\MessageBreak when you use the optional argument
3754 of \string\makeglossaries}{Either move your definitions to the
3755 preamble or don't use the optional argument of
3756 \string\makeglossaries}%
3757 }%

```

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).

```

3758 \let\@glo@assign@sortkey\@glxtr@mixed@assign@sortkey
3759 \renewcommand*\@printgloss@setsort{%

```

Need to extract just the type value.

```

3760 \expandafter\@glxtr@gettype\expandafter,\@glxtr@printglossopts,%
3761 type=\glsdefaulttype,\@end@glxtr@gettype
3762 \def\@glo@sorttype{\@glo@default@sorttype}%
3763 }%

```

Check automake setting:

```

3764 \ifglautomake
3765 \renewcommand*\@gls@doautomake{%
3766 \@for\@gls@type:=\@glxtr@reg@glosslist\do{%
3767 \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3768 }%
3769 }%
3770 \fi

```

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

```

3771 \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
3772 \fi
3773 }%
3774 \fi
3775 }

```

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

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

```

3776 \newcommand{\@glxtr@orgprintglossary}[2]{%
3777 \def\@glo@type{\glsdefaulttype}%

```

Add check here.

```

3778 \def\glossarytitle{%
3779 \ifcsdef{\@glo@type@\@glo@type @title}%
3780 {\csuse{\@glo@type@\@glo@type @title}}%
3781 {\glossaryname}}%
3782 \def\glossarytoctitle{\glossarytitle}%

```

```

3783 \let\org@glossarytitle\glossarytitle
3784 \def\@glossarystyle{%
3785   \ifx\@glossary@default@style\relax
3786     \GlossariesWarning{No default glossary style provided \MessageBreak
3787       for the glossary '\@glo@type'. \MessageBreak
3788       Using deprecated fallback. \MessageBreak
3789       To fix this set the style with \MessageBreak
3790       \string\setglossarystyle\space or use the \MessageBreak
3791       style key=value option}%
3792   \fi
3793 }%
3794 \def\gls@dotoc@title{\gls@settoc@title{\@glo@type}}%
3795 \let\org@glossaryentrynumbers\glossaryentrynumbers
3796 \bgroup
3797   \@printgloss@setsort
3798   \setkeys{printgloss}{#1}%
3799   \ifx\glossarytitle\org@glossarytitle
3800   \else
3801     \cslet{\@glo@type@\@glo@type @title}{\glossarytitle}%
3802   \fi
3803   \let\currentglossary\@glo@type
3804   \let\org@glossaryentrynumbers\glossaryentrynumbers
3805   \let\glsnonextpages\@glsnonextpages
3806   \let\glsnextpages\@glsnextpages

3807   \glsxtractivatenopost
3808   \gls@dotoc@title
3809   \@glossarystyle
3810   \let\gls@org@glossaryentryfield\glossentry
3811   \let\gls@org@glossarysubentryfield\subglossentry
3812   \renewcommand{\glossentry}[1]{%
3813     \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3814     \gls@org@glossaryentryfield{##1}%
3815   }%
3816   \renewcommand{\subglossentry}[2]{%
3817     \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3818     \gls@org@glossarysubentryfield{##1}{##2}%
3819   }%
3820   \@gls@preglossaryhook
3821   #2%
3822 \egroup
3823 \global\let\glossaryentrynumbers\org@glossaryentrynumbers
3824 \global\let\warn@noprintglossary\relax
3825 }

```

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

```

3826 \newcommand*\glsxtractivatenopost{%
3827   \let\nopostdesc\@nopostdesc
3828   \let\glsxtrnopostpunc\@glsxtr@nopostpunc
3829 }

```

lsxtrnopostpunc

```
3830 \newrobustcmd*{\glxtrnopostpunc}{}
```

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

```
3831 \newcommand{\@glxtr@nopostpunc}{%
3832 \let\@glxtr@org@postdescription\glspostdescription
3833 \ifglsnopostdot
3834 \renewcommand{\glspostdescription}{%
3835 \glsnopostdottrue
3836 \let\glspostdescription\@glxtr@org@postdescription
3837 \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3838 \glxtrpostdescription
3839 \@glxtr@nopostpunc@postdesc}%
3840 \else
3841 \renewcommand{\glspostdescription}{%
3842 \let\glspostdescription\@glxtr@org@postdescription
3843 \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3844 \glxtrpostdescription
3845 \@glxtr@nopostpunc@postdesc}%
3846 \fi
3847 \glsnopostdotfalse
3848 }
```

stpunc@postdesc

```
3849 \newcommand*{\@glxtr@nopostpunc@postdesc}{}
```

estore@postpunc

```
3850 \newcommand*{\@glxtr@restore@postpunc}{%
3851 \def\@glxtr@nopostpunc@postdesc{%
3852 \@glxtr@org@postdescription
3853 \let\@glxtr@nopostpunc@postdesc\@empty
3854 \let\glxtrrestorepostpunc\@empty
3855 }%
3856 }
```

restorepostpunc Does nothing outside of glossary.

```
3857 \newcommand*{\glxtrrestorepostpunc}{}
```

\@printglossary Redefine.

```
3858 \renewcommand{\@printglossary}[2]{%
3859 \def\@glxtr@printglossopts{#1}%
3860 \@glxtr@orgprintglossary{#1}{#2}%
3861 }
```

Add a key that switches off the entry targets:

```
3862 \define@choicekey{printgloss}{target}
3863 [\@glxtr@printglossval\@glxtr@printglossnr]%
```

```

3864 {true,false}[true]%
3865 {%
3866   \ifcase\@glxtr@printglossnr
3867     \let\@glstarget\glsdohypertarget
3868   \else
3869     \let\@glstarget\@secondoftwo
3870   \fi
3871 }

```

hypernameprefix

```

3872 \newcommand{\@glxtrhypernameprefix}{}

```

New to v1.20:

```

3873 \define@key{printgloss}{targetnameprefix}{%
3874   \renewcommand{\@glxtrhypernameprefix}{#1}%
3875 }

```

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

```

3876 \let\@glxtr@org@glsdohypertarget\glsdohypertarget
3877 \renewcommand{\glsdohypertarget}[2]{%
3878   \@glxtr@org@glsdohypertarget{\@glxtrhypernameprefix#1}{#2}%
3879 }

```

@makeglossaries For the benefit of makeglossaries

```

3880 \newcommand*{\glxtr@makeglossaries}[1]{}

```

@glxtr@gettype Get just the type.

```

3881 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
3882   \def\@glo@type{#2}%
3883 }

```

@assign@sortkey Assign the sort key.

```

3884 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
3885   \edef\@glo@type{\@glo@type}%
3886   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
3887   {%
3888     \@glo@no@assign@sortkey{#1}%
3889   }%
3890   {%
3891     \@glo@assign@sortkey{#1}%
3892   }%
3893 }%

```

Display number list for the regular version:

splaynumberlist

```

3894 \let\@glxtr@idx@displaynumberlist\glsdisplaynumberlist

```

Display number list for the “noidx” version:

splaynumberlist

```

3895 \newcommand*{\@glsxtr@noidx@displaynumberlist}[1]{%
3896   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3897   \ifdef\@gls@loclist
3898   {%
3899     \def\@gls@noidxloclist@sep{%
3900       \def\@gls@noidxloclist@sep{%
3901         \def\@gls@noidxloclist@sep{%
3902           \glsnumlistsep
3903         }%
3904       \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
3905     }%
3906   }%
3907   \def\@gls@noidxloclist@finalsep{}%
3908   \def\@gls@noidxloclist@prev{}%
3909   \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
3910   \@gls@noidxloclist@finalsep
3911   \@gls@noidxloclist@prev
3912 }%
3913 {%
3914   \glsxtrundeftag
3915   \glsdoifexists{#1}%
3916   {%
3917     \GlossariesWarning{Missing location list for ‘#1’. Either
3918       a rerun is required or you haven’t referenced the entry.}%
3919   }%
3920 }%
3921 }%
3922

```

And for the number list loop:

@numberlistloop

```

3923 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
3924   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3925   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
3926   \let\@gls@org@glsseeformat\glsseeformat
3927   \let\glsnoidxdisplayloc#2\relax
3928   \let\glsseeformat#3\relax
3929   \ifdef\@gls@loclist
3930   {%
3931     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3932   }%
3933   {%
3934     \glsxtrundeftag
3935     \glsdoifexists{#1}%
3936     {%
3937       \GlossariesWarning{Missing location list for ‘##1’. Either

```

```

3938         a rerun is required or you haven't referenced the entry.}%
3939     }%
3940 }%
3941 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3942 \let\glsseeformat\@gls@org@glsseeformat
3943 }%

```

Same for entry number list.

entrynumberlist

```

3944 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
3945   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3946   \ifdef\@gls@loclist
3947   {%
3948     \glsnoidxloclist{\@gls@loclist}%
3949   }%
3950   {%
3951     \glsxtrundeftag
3952     \glsdoifexists{#1}%
3953     {%
3954       \GlossariesWarning{Missing location list for '#1'. Either
3955         a rerun is required or you haven't referenced the entry.}%
3956     }%
3957   }%
3958 }%

```

entrynumberlist

```

3959 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

3960 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
3961   \protected@edef\@glsxtr@titlelabel{#1}%
3962   \ifdefvoid\@glsxtr@titlelabel
3963   {%
3964   {%
3965     \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle{#1}}}%
3966   }%
3967   \ifdefvoid{\@glsxtr@titlelabel}%
3968   {%
3969     \DTLifint{#1}%
3970   {%
3971     \ifnum#1<256\relax
3972     \edef#2{\char#1\relax}%
3973   \else
3974     \edef#2{#1}%
3975   \fi
3976   }%
3977   {%
3978     \ifcsundef{#1groupname}%

```

```

3979      {\def#2{#1}}%
3980      {\letcs#2{#1groupname}}}%
3981  }%
3982 }%
3983 {%
3984   \let#2\@glsxtr@titlelabel
3985 }%
3986 }

```

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

```

3987 \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.

```

3988 \newrobustcmd{\glsxtrgetgrouptitle}[2]{%
3989   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3990   \@onelevel@sanitize\@glsxtr@titlelabel
3991   \ifcsdef{\@glsxtr@titlelabel}
3992     {\letcs{#2}{\@glsxtr@titlelabel}}%
3993     {\glsxtr@org@getgrouptitle{#1}{#2}}%
3994 }
3995 \let\@gls@getgrouptitle\glsxtrgetgrouptitle

```

trsetgrouptitle Sets the title for the given group label.

```

3996 \newcommand{\glsxtrsetgrouptitle}[2]{%
3997   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3998   \@onelevel@sanitize\@glsxtr@titlelabel
3999   \protected@csxdef{\@glsxtr@titlelabel}{#2}%
4000 }

```

alsetgrouptitle As above put only locally defines the title.

```

4001 \newcommand{\glsxtrlocalsetgrouptitle}[2]{%
4002   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
4003   \@onelevel@sanitize\@glsxtr@titlelabel
4004   \protected@csedef{\@glsxtr@titlelabel}{#2}%
4005 }

```

\glsnavigation Redefine to use new user-level command.

```

4006 \renewcommand*{\glsnavigation}{%
4007   \def\@gls@between{}%
4008   \ifcsundef{\@gls@hypergroupelist@\@glo@type}%
4009   {%
4010     \def\@gls@list{}%
4011   }%
4012   {%
4013     \expandafter\let\expandafter\@gls@list
4014       \csname \@gls@hypergroupelist@\@glo@type\endcsname
4015   }%

```

```

4016 \@for\@gls@tmp:=\@gls@list\do{%
4017   \@gls@between
4018   \glsxtrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
4019   \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
4020   \let\@gls@between\glshypernavsep
4021 }%
4022 }

```

@noidx@glossary

```

4023 \renewcommand*{\@print@noidx@glossary}{%
4024   \ifcsdef{\glsref@{\@glo@type}%
4025   {%
4026     \ifcsdef{\@glo@sortmacro@\@glo@sorttype}%
4027     {%
4028       \csuse{\@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
4029     }%
4030     {%
4031       \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
4032     }%
4033     \glossarysection[\glossarytoctitle]{\glossarytitle}%
4034     \glossarypreamble

```

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

```

4035   \def\@gls@currentlettergroup{%
4036     \begin{theglossary}%
4037     \glossaryheader
4038     \glsresetentrylist
4039     \forlistcsloop{\@gls@noidx@do}{\glsref@{\@glo@type}%
4040     \end{theglossary}%
4041     \glossarypostamble
4042   }%
4043   {%

```

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

```

4044   \glsxtrifemptyglossary{\@glo@type}%
4045   }%
4046   {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
4047   \@gls@noref@warn{\@glo@type}%
4048 }%
4049 }

```

noidxdisplayloc Patch to check for range formations.

```

4050 \renewcommand*{\glsnoidxdisplayloc}[4]{%
4051   \setentrycounter[#1]{#2}%
4052   \@glsxtr@display@loc#3\empty\end@glxtr@display@loc{#4}%
4053 }

```


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

```
4054 \def\@glxtr@display@loc#1#2\end@glxtr@display@loc#3{%
4055   \ifx#1\relax
4056     \glxtrdisplaystartloc{#2}{#3}%
4057   \else
4058     \ifx#1)\relax
4059       \glxtrdisplayendloc{#2}{#3}%
4060     \else
4061       \glxtrdisplaysingleloc{#1#2}{#3}%
4062     \fi
4063   \fi
4064 }
```

isplaysingleloc Single location.

```
4065 \newcommand*\glxtrdisplaysingleloc}[2]{%
4066   \csuse{#1}{#2}%
4067 }
```

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

displaystartloc Start of a location range.

```
4068 \newcommand*\glxtrdisplaystartloc}[2]{%
4069   \edef\glxtrlocrangefmt{#1}%
4070   \ifx\glxtrlocrangefmt\empty
4071     \def\glxtrlocrangefmt{glnumberformat}%
4072   \fi
4073   \expandafter\glxtrdisplaysingleloc
4074   \expandafter{\glxtrlocrangefmt}{#2}%
4075 }
```

trdisplayendloc End of a location range.

```
4076 \newcommand*\glxtrdisplayendloc}[2]{%
4077   \edef\@glxtr@tmp{#1}%
4078   \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{glnumberformat}}{}%
4079   \ifx\glxtrlocrangefmt\@glxtr@tmp
4080   \else
4081     \GlossariesExtraWarning{Mismatched end location range
4082       (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
4083   \fi
4084   \expandafter\glxtrdisplayendloohook\expandafter{\@glxtr@tmp}{#2}%
4085   \expandafter\glxtrdisplaysingleloc
4086   \expandafter{\glxtrlocrangefmt}{#2}%
4087   \def\glxtrlocrangefmt{}%
4088 }
```

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

```
4089 \newcommand*\glxtrdisplayendloohook}[2]{}
```

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

```
4090 \newcommand*{\gsxtrlocrangefmt}{}%
```

gssetentrycounter Adjust \setentrycounter to save the original prefix.

```
4091 \renewcommand*{\setentrycounter}[2] [] {%
4092   \def\gsxtrcounterprefix{#1}%
4093   \ifx\gsxtrcounterprefix\@empty
4094     \def\@glo@counterprefix{.}%
4095   \else
4096     \def\@glo@counterprefix{.#1.}%
4097   \fi
4098   \def\glsentrycounter{#2}%
4099 }
```

gs@removespaces Redefine to allow adjustments to location hyperlink.

```
4100 \def\@gls@removespaces#1 #2\@nil{%
4101   \toks@=\expandafter{\the\toks@#1}%
4102   \ifx\@#2\@nil
4103     \edef\x{\the\toks@}%
4104     \ifx\x\empty
4105     \else
```

Expand location (just in case \toks@ is needed for something else).

```
4106     \expandafter\gsxtrlocationhyperlink\expandafter
4107     \glsentrycounter\expandafter\@glo@counterprefix\expandafter{\the\toks@}%
4108   \fi
4109   \else
4110     \@gls@ReturnAfterFi{%
4111       \@gls@removespaces#2\@nil
4112     }%
4113   \fi
4114 }
```

gslocationhyperlink

```
4115 \newcommand*{\gsxtrlocationhyperlink}[3] {%
4116   \ifdefined\gsxtrsupplocationurl
4117   {%
4118     \GlsXtrInternalLocationHyperlink{#1}{#2}{#3}%
4119   }%
4120   {%
4121     \hyperref{\gsxtrsupplocationurl}{#1#2#3}{#3}%
4122   }%
4123 }
```

gssupphypernumber

```
4124 \newcommand*{\gsxtrsupphypernumber}[1] {%
4125   {%
4126     \glshasattribute{\glscurrententrylabel}{externallocation}%
4127     {%
```

```

4128     \def\glxtrsupplocationurl{%
4129         \glxgetattribute{\glscurrententrylabel}{externallocation}}%
4130     }%
4131     {%
4132     \def\glxtrsupplocationurl{%
4133     }%
4134     \glshypernumber{#1}%
4135     }%
4136 }

```

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

@print@glossary

```

4137 \renewcommand{\@print@glossary}{%
4138     \makeatletter
4139     \@input{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
4140     \IfFileExists{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
4141     {}%
4142     {\glxtrNoGlossaryWarning{\@glo@type}}%
4143     \ifglxindy
4144         \ifcsundef{\xdy\@glo@type @language}%
4145         {%
4146             \edef\@do@auxoutstuff{%
4147                 \noexpand\AtEndDocument{%
4148                     \noexpand\immediate\noexpand\write\@auxout{%
4149                         \string\providecommand\string\@xdylanguage[2]{}}%
4150                     \noexpand\immediate\noexpand\write\@auxout{%
4151                         \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4152                     }%
4153                 }%
4154             }%
4155             {%
4156                 \edef\@do@auxoutstuff{%
4157                     \noexpand\AtEndDocument{%
4158                         \noexpand\immediate\noexpand\write\@auxout{%
4159                             \string\providecommand\string\@xdylanguage[2]{}}%
4160                         \noexpand\immediate\noexpand\write\@auxout{%
4161                             \string\@xdylanguage{\@glo@type}{\csname @xdy\@glo@type
4162                                 @language\endcsname}}%
4163                         }%
4164                     }%
4165                 }%
4166             \@do@auxoutstuff
4167             \edef\@do@auxoutstuff{%
4168                 \noexpand\AtEndDocument{%
4169                     \noexpand\immediate\noexpand\write\@auxout{%
4170                         \string\providecommand\string\@gls@codepage[2]{}}%
4171                     \noexpand\immediate\noexpand\write\@auxout{%
4172                         \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%

```

```

4173     }%
4174     }%
4175     \@do@auxoutstuff
4176 \fi
4177 \renewcommand*{\@warn@nomakeglossaries}{%
4178     \GlossariesWarningNoLine{\string\makeglossaries\space
4179     hasn't been used,^^Jthe glossaries will not be updated}%
4180 }%
4181 }

```

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

`\GlsWarningHead` Header message.

```

4182 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
4183     This document is incomplete. The external file associated with
4184     the glossary '#1' (which should be called \texttt{#2})
4185     hasn't been created.%
4186 }

```

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

```

4187 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
4188     This has probably happened because there are no entries defined
4189     in this glossary.%
4190 }

```

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

```

4191 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
4192     If you don't want this glossary,
4193     add \texttt{nomain} to your package option list when you load
4194     \texttt{glossaries-extra.sty}. For example:%
4195 }

```

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

```

4196 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
4197     Did you forget to use \texttt{type=#1} when you defined your
4198     entries? If you tried to load entries into this glossary with
4199     \texttt{\string\loadglsentries} did you remember to use
4200     \texttt{[#1]} as the optional argument? If you did, check that
4201     the definitions in the file you loaded all had the type set
4202     to \texttt{\string\glsdefaulttype}.%
4203 }

```

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

```

4204 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
4205     Check the contents of the file \texttt{#1}. If
4206     it's empty, that means you haven't indexed any of your entries in this
4207     glossary (using commands like \texttt{\string\gls} or
4208     \texttt{\string\glsadd}) so this list can't be generated.
4209     If the file isn't empty, the document build process hasn't been

```

```

4210 completed.%
4211 }

```

WarningAutoMake Message when automake option has been used.

```

4212 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
4213   You may need to rerun \LaTeX. If you already have, it may be that
4214   \TeX's shell escape doesn't allow you to run
4215   \ifglxindy xindy\else makeindex\fi. Check the
4216   transcript file \texttt{\jobname.log}. If the shell escape is
4217   disabled, try one of the following:
4218
4219   \begin{itemize}
4220     \item Run the external (Lua) application:
4221
4222       \texttt{makeglossaries-lite.lua \string\jobname\string}
4223
4224     \item Run the external (Perl) application:
4225
4226       \texttt{makeglossaries \string\jobname\string}
4227   \end{itemize}
4228
4229   Then rerun \LaTeX\ on this document.
4230   \GlossariesExtraWarning{Rerun required to build the
4231   glossary '#1' or check TeX's shell escape allows
4232   you to run \ifglxindy xindy\else makeindex\fi}%
4233 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

4234 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
4235   You need to either replace \texttt{\string\makenoidxglossaries}
4236   with \texttt{\string\makeglossaries} or replace
4237   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
4238   \texttt{\string\printnoidxglossary}
4239   (or \texttt{\string\printnoidxglossaries}) and then rebuild
4240   this document.%
4241 }

```

WarningBuildInfo Build advice.

```

4242 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
4243   Try one of the following:
4244   \begin{itemize}
4245     \item Add \texttt{automake} to your package option list when you load
4246       \texttt{glossaries-extra.sty}. For example:
4247
4248       \texttt{\string\usepackage[automake]%
4249       \glsopenbrace glossaries-extra\glsclosebrace}
4250
4251     \item Run the external (Lua) application:

```

```

4252
4253     \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4254
4255     \item Run the external (Perl) application:
4256
4257     \texttt{makeglossaries \string"\jobname\string"}
4258 \end{itemize}
4259
4260 Then rerun \LaTeX\ on this document.%
4261 }

```

oGlsWarningTail Final paragraph.

```

4262 \newcommand{\GlsXtrNoGlsWarningTail}{%
4263 This message will be removed once the problem has been fixed.%
4264 }

```

GlsWarningNoOut No out file created. Build advice.

```

4265 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
4266 The file \texttt{#1} doesn't exist. This most likely means you haven't used
4267 \texttt{\string\makeglossaries} or you have used
4268 \texttt{\string\nofiles}. If this is just a draft version of the
4269 document, you can suppress this message using the
4270 \texttt{nomissingglstext} package option.%
4271 }

```

glossarywarning

```

4272 \newcommand*{@@glsxtr@defaultnoglossarywarning}[1]{%
4273 \glossarysection[\glossarytoctitle]{\glossarytitle}
4274 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glo@type @in\endcsname}
4275 \par
4276 \glsxtrifemptyglossary{#1}%
4277 {%
4278 \GlsXtrNoGlsWarningEmptyStart\space
4279 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
4280 \medskip
4281 \noindent\texttt{\string\usepackage[nomain\ifglsacronym ,acronym\fi]%
4282 \glsopenbrace glossaries-extra\glsclosebrace}
4283 \medskip
4284 }%
4285 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
4286 }%
4287 {%
4288 \IfFileExists{\jobname.\csname @glo@type @\@glo@type @out\endcsname}
4289 {%
4290 \GlsXtrNoGlsWarningCheckFile
4291 {\jobname.\csname @glo@type @\@glo@type @out\endcsname}
4292
4293 \ifglsautomake
4294

```

```

4295     \GlsXtrNoGlsWarningAutoMake{#1}
4296
4297     \else
4298
4299         \ifthenelse{\equal{#1}{main}}{%
4300             {%
4301                 \GlsXtrNoGlsWarningEmptyMain\par
4302                 \medskip
4303                 \noindent\texttt{\string\usepackage[nomain]%
4304                     \glsopenbrace glossaries-extra\glsclosebrace}
4305                 \medskip
4306             }%
4307         }%
4308
4309         \ifdefequal\makeglossaries\@no@makeglossaries
4310             {%
4311                 \GlsXtrNoGlsWarningMisMatch
4312             }%
4313             {%
4314                 \GlsXtrNoGlsWarningBuildInfo
4315             }%
4316         \fi
4317     }%
4318     {%
4319         \GlsXtrNoGlsWarningNoOut
4320         {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
4321     }%
4322 }%
4323 \par
4324 \GlsXtrNoGlsWarningTail
4325 }

```

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

xttrresourcefile 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.

```

4326 \newcommand*{\glxtrresourcefile}[2] [] {%

```

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

```

4327     \disable@keys{glossaries-extra.sty}{record}%
4328     \glxtr@writefields
4329     \protected@write\@auxout{\glxtrresourceinit}{\string\glxtr@resource{#1}{#2}}%
4330     \let\@glxtr@org@see@noindex\@gls@see@noindex
4331     \let\@gls@see@noindex\relax
4332     \IfFileExists{#2.glstex}%
4333     {%

```

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

```

4334     \edef\@bibgls@restoreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%

```

```

4335 \makeatletter
4336 \@input{#2.glstex}%
4337 \@bibgls@restoreat
4338 }%
4339 {%
4340 \GlossariesExtraWarning{No file ‘#2.glstex’}%
4341 }%
4342 \let\@gls@see@noindex\@glsxtr@org@see@noindex
4343 }
4344 \@onlypreamble\glsxtrresourcefile

xtrresourceinit Code used during the protected write operation.
4345 \newcommand*{\glsxtrresourceinit}{%

trresourcecount
4346 \newcount\glsxtrresourcecount

trLoadResources Short cut that uses \glsxtrresourcefile with \jobname as the mandatory argument.
4347 \newcommand*{\GlsXtrLoadResources}[1][1]{%
4348 \ifnum\glsxtrresourcecount=0\relax
4349 \glsxtrresourcefile[#1]{\jobname}%
4350 \else
4351 \glsxtrresourcefile[#1]{\jobname-\the\glsxtrresourcecount}%
4352 \fi
4353 \advance\glsxtrresourcecount by 1\relax
4354 }

glsxtr@resource
4355 \newcommand*{\glsxtr@resource}[2]{%

\glsxtr@fields
4356 \newcommand*{\glsxtr@fields}[1]{%

xtr@texencoding
4357 \newcommand*{\glsxtr@texencoding}[1]{%

\glsxtr@langtag
4358 \newcommand*{\glsxtr@langtag}[1]{%

@pluralsuffixes
4359 \newcommand*{\glsxtr@pluralsuffixes}[4]{%

tr@shortcutsval
4360 \newcommand*{\glsxtr@shortcutsval}[1]{%

sxtr@linkprefix
4361 \newcommand*{\glsxtr@linkprefix}[1]{%

```


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

```
4362 \newcommand*{\glxtr@writefields}{%
4363   \protected@write\@auxout{}%
4364     {\string\providecommand*{\string\glxtr@fields}[1]{}}%
4365   \protected@write\@auxout{}%
4366     {\string\providecommand*{\string\glxtr@resource}[2]{}}%
4367   \protected@write\@auxout{}%
4368     {\string\providecommand*{\string\glxtr@pluralsuffixes}[4]{}}%
4369   \protected@write\@auxout{}%
4370     {\string\providecommand*{\string\glxtr@shortcutsval}[1]{}}%
4371   \protected@write\@auxout{}%
4372     {\string\providecommand*{\string\glxtr@linkprefix}[1]{}}%
4373   \protected@write\@auxout{}{\string\glxtr@fields{\@glx@keymap}}%

4374   \protected@write\@auxout{}%
4375     {\string\providecommand*{\string\glxtr@record}[5]{}}%
```

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

```
4376 \ifdef\CurrentTrackedLanguageTag
4377 {%
4378   \protected@write\@auxout{}%
4379     {\string\glxtr@langtag{\CurrentTrackedLanguageTag}}%
4380 }%
4381 {}%
4382 \protected@write\@auxout{}{\string\glxtr@pluralsuffixes
4383   {\glxpluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
4384   {\glxtrabbrvpluralsuffix}}%
4385 \ifdef\inputencodingname
4386 {%
4387   \protected@write\@auxout{}{\string\glxtr@texencoding{\inputencodingname}}%
4388 }%
4389 {}%
```

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.)

```
4390   \@ifpackageloaded{fontspec}%
4391     {\protected@write\@auxout{}{\string\glxtr@texencoding{utf8}}}%
4392   {}%
4393 }%
4394 \protected@write\@auxout{}{\string\glxtr@shortcutsval{\@glxtr@shortcutsval}}%
```

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

```
4395 \AtBeginDocument
4396   {\protected@write\@auxout{}{\string\glxtr@linkprefix{\glolinkprefix}}}%
4397 \let\glxtr@writefields\relax
```

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

```
4398 \ifglautomake
4399 \IfFileExists{\jobname.aux}%
4400 {\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.

```
4401 \ifx\@gls@doautomake\@gls@doautomake@err
4402 \let\@gls@doautomake\relax
4403 \fi
4404 \fi
4405 }
```

do@automake@err

```
4406 \newcommand*{\@gls@doautomake@err}{%
4407 \PackageError{glossaries}{You must use
4408 \string\makeglossaries\space with automake=true}
4409 {%
4410 Either remove the automake=true setting or
4411 add \string\makeglossaries\space to your document preamble.%
4412 }%
4413 }
```

Allow locations specific to a particular counter to be recorded.

\glsxtr@record

```
4414 \newcommand*{\glsxtr@record}[5] {}
```

r@counterrecord Aux file command.

```
4415 \newcommand*{\glsxtr@counterrecord}[3]{%
4416 \glsxtrfieldlistgadd{#1}{record.#2}{#3}%
4417 }
```

unterrecordhook Hook used by \@glsxtr@dorecord.

```
4418 \newcommand*{\@glsxtr@counterrecordhook} {}
```

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

```
4419 \newcommand*{\GlsXtrRecordCounter}[1]{%
4420 \@glsxtr@recordcounter{#1}%
4421 }
4422 \@onlypreamble\GlsXtrRecordCounter
```

docounterrecord

```
4423 \newcommand*{\@glsxtr@docounterrecord}[1]{%
4424 \protected@write\@auxout{}{\string\glsxtr@counterrecord
4425 {\@gls@label}{#1}{\csuse{the#1}}}%
4426 }
```

`\lsxtrglossentry` Users may prefer to have entries displayed throughout the document rather than gathered together in a list. This command emulates the way `\glossentry` behaves (without the style formatting commands like `\item`). This needs to define `\currentglossary` to the current glossary type (normally set at the start of `\@printglossary`) and needs to define `\glscurrententrylabel` to the entry's label (normally set before `\glossentry` and `\subglossentry`). This needs some protection in case it's used in a section heading.

```

4427 \newcommand*\lsxtrglossentry}[1]{%
4428   \glstrtitleorpdforheading
4429   {\@lsxtrglossentry{#1}}%
4430   {\glsentryname{#1}}%
4431   {\glsxtrheadname{#1}}%
4432 }

```

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

```

4433 \newrobustcmd*\@lsxtrglossentry}[1]{%
4434   \glstrtitleorpdforheading
4435   {%
4436     \glsdexists{#1}%
4437     {%
4438       \begingroup
4439       \edef\glscurrententrylabel{\glsdetoklabel{#1}}%
4440       \edef\currentglossary{\glsentrytype{\glscurrententrylabel}}%
4441       \ifglshasparent{#1}%
4442       {\glssubentryitem{#1}}%
4443       {\glsentryitem{#1}}%
4444       \glstarget{#1}{\glossentryname{#1}}%
4445     \endgroup
4446   }%
4447 }%
4448 {\glsentryname{#1}}%
4449 {\glsxtrheadname{#1}}%
4450 }

```

`\glossentryother` As `\lsxtrglossentry` 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.

```

4451 \newcommand*\glossentryother}[3]{%
4452   \ifstrempy{#1}%
4453   {%
4454     \ifcsdef{glsxtrhead#3}%
4455     {%
4456       \glstrtitleorpdforheading
4457       {\@lsxtrglossentryother{#2}{#3}{#1}}%
4458       {\@glse@entry@field{#2}{#3}}%
4459       {\csuse{glsxtrhead#3}{#2}}%
4460     }%

```

```

4461  {%
4462    \glstrtitleorpdforheading
4463    {\@glstrglossentryother{#2}{#3}{#1}}%
4464    {\@glentry@field{#2}{#3}}%
4465    {\@glentry@field{\NoCaseChange{#2}}{#3}}%
4466  }%
4467 }%
4468 {%
4469   \glstrtitleorpdforheading
4470   {\@glstrglossentryother{#2}{#3}{#1}}%
4471   {\@glentry@field{#2}{#3}}%
4472   {#1}%
4473 }%
4474 }

```

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

```

4475 \newrobustcmd*{\@glstrglossentryother}[3]{%
4476   \glstrtitleorpdforheading
4477   {%
4478     \glsoifexists{#1}%
4479     {%
4480       \beginngroup
4481         \edef\glscurrententrylabel{\glsetoklabel{#1}}%
4482         \edef\currentglossary{\glentrytype{\glscurrententrylabel}}%
4483         \ifglshasparent{#1}%
4484           {\glssubentryitem{#1}}%
4485           {\glentryitem{#1}}%
4486           \glstarget{#1}{\glossentrynameother{#1}{#2}}%
4487       \endgroup
4488     }%
4489   }%
4490   {\@glentry@field{#1}{#2}}%
4491   {#3}%
4492 }

```

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

```

4493 \newcommand*{\printunsrtglossary}{%
4494   \@ifstar\s@printunsrtglossary\@printunsrtglossary
4495 }

```

`ntunsrtglossary` Unstarred version.

```

4496 \newcommand*{\@printunsrtglossary}[1][ ]{%
4497   \@printglossary{type=\gldefaulttype,#1}{\@print@unsrt@glossary}%
4498 }

```

`ntunsrtglossary` Starred version.

```

4499 \newcommand*{\s@printunsrtglossary}[2][ ]{%
4500   \beginngroup

```

```

4501     #2%
4502     \@printglossary{type=\glsdefaultttype,#1}{\@print@unsrt@glossary}%
4503 \endgroup
4504 }

```

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

```

4505 \newcommand*{\printunsrtglossaries}{%
4506   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
4507 }

```

`\@unsrt@glossary`

```

4508 \newcommand*{\@print@unsrt@glossary}{%
4509   \glossarysection[\glossarytoctitle]{\glossarytitle}%
4510   \glossarypreamble
4511   check for empty list
4512   \glsxtrifemptyglossary{\@glo@type}%
4513   {%
4514     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
4515   }%
4516   {%
4517     \key@ifundefined{glossentry}{group}%
4518     {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
4519     {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
4520     \def\@gls@currentlettergroup{}}%

```

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

```

4520 \def\@glsxtr@doglossary{%
4521   \begin{theglossary}%
4522   \glossaryheader
4523   \glsresetentrylist
4524 }%
4525 \expandafter\@for\expandafter\@gls@currententrylabel\expandafter
4526   :\expandafter=\csname glo@list@\@glo@type\endcsname\do{%
4527   \ifdefempty{\@gls@currententrylabel}
4528   {}%
4529   {%

```

Provide a hook (for example to measure width).

```

4530   \let\@glsxtr@process\@firstofone
4531   \let\printunsrtglossaryskipentry
4532     \@glsxtr@printunsrtglossaryskipentry
4533   \printunsrtglossaryentryprocesshook{\@gls@currententrylabel}%

```

Don't check group for child entries.

```

4534   \glsxtr@process
4535   {%
4536     \ifglshasparent{\@gls@currententrylabel}{}%
4537     {%

```

```

4538         \@glxtr@checkgroup\glscurrententrylabel
4539         \expandafter\appto\expandafter\@glxtr@doglossary\expandafter
4540         {\@glxtr@groupheading}%
4541     }%
4542     \eappto\@glxtr@doglossary{%
4543         \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
4544     }%
4545 }%
4546 }%
4547 \appto\@glxtr@doglossary{\end{theglossary}}%
4548 \printunsrtglossarypredoglossary
4549 \@glxtr@doglossary
4550 }%
4551 \glossarypostamble
4552 }

```

entryprocesshook

```

4553 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{%

```

entryprocesshook

```

4554 \newcommand*{\printunsrtglossaryskipentry}{%
4555     \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
4556 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
4557 }

```

entryprocesshook

```

4558 \newcommand*{\@glxtr@printunsrtglossaryskipentry}{%
4559     \let\glxtr@process\gobble
4560 }

```

rypredoglossary

```

4561 \newcommand*{\printunsrtglossarypredoglossary}{%

```

lossary@handler

```

4562 \newcommand{\@printunsrt@glossary@handler}[1]{%
4563     \xdef\glscurrententrylabel{#1}%
4564     \printunsrtglossaryhandler\glscurrententrylabel
4565 }

```

glossaryhandler

```

4566 \newcommand{\printunsrtglossaryhandler}[1]{%
4567     \glxtrunsrtdo{#1}%
4568 }

```

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.

```

4569 \newrobustcmd*{\glxtriflabelinlist}[4]{%

```

```

4570 \protected@edef\@glxstr@doiflabelinlist{\noexpand\@glx@ifinlist{#1}{#2}}%
4571 \@glxstr@doiflabelinlist{#3}{#4}%
4572 }

```

srtglossaryunit

```

4573 \newcommand{\print@op@unsrtglossaryunit}[2][]{%
4574   \s@printunsrtglossary[type=\glxdefaulttype,#1]{%
4575     \printunsrtglossaryunitsetup{#2}%
4576   }%
4577 }

```

ossaryunitsetup

```

4578 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
4579   \renewcommand{\printunsrtglossaryhandler}[1]{%
4580     \glxstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
4581     {\glxstrunsrtdo{##1}}%
4582   }%
4583 }%

```

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

```

4584 \ifcsundef{theH#1}%
4585 {%
4586   \renewcommand*{\@glxstrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
4587 }%
4588 {%
4589   \renewcommand*{\@glxstrhypernameprefix}{record.#1.\csuse{theH#1}.\@gobble}%
4590 }%
4591 \renewcommand*{\glossarysection}[2][]{}%
4592 \appto\glossarypostamble{\glxpar\medskip\glxpar}%
4593 }

```

srtglossaryunit

```

4594 \newcommand{\print@noop@unsrtglossaryunit}[2][]{%
4595   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
4596     requires the record=only or record=alsoindex package option}{}%
4597 }

```

t@getgrouptitle

```

4598 \newrobustcmd*{\@glxstr@unsrt@getgrouptitle}[2]{%
4599   \protected@edef\@glxstr@titlelabel{\glxstr@grouptitle@#1}%
4600   \onelevel@sanitize\@glxstr@titlelabel
4601   \ifcsdef{\@glxstr@titlelabel}
4602     {\letcs{#2}{\@glxstr@titlelabel}}%
4603     {\def#2{#1}}%
4604 }

```

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

```

4605 \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).

```
4606 \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.

```
4607 \newcommand*{\@glsxtr@checkgroup}[1]{%
4608   \def\@glsxtr@groupheading{}%
4609   \key@ifundefined{glossentry}{group}%
4610   {%
4611     \letcs{\@gls@sort}{glo\@glsdetoklabel{#1}@sort}%
4612     \expandafter\glo@grabfirst\@gls@sort{}\@nil
4613   }%
4614   {%
4615     \protected@edef\@glo@thislettergrp{%
4616       \csuse{glo\@glsdetoklabel{#1}\@glsxtrgroupfield}}%
4617     }%
4618     \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4619     {}%
4620     {%
4621       \ifdefempty{\@gls@currentlettergroup}{}%
4622       {\def\@glsxtr@groupheading{\gls@groupskip}}%
4623       \eappto\@glsxtr@groupheading{%
4624         \noexpand\gls@groupheading{\expandonce\@glo@thislettergrp}%
4625       }%
4626     }%
4627     \let\@gls@currentlettergroup\@glo@thislettergrp
4628 }
```

`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.

```
4629 \newcommand{\@glsxtr@noidx@do}[1]{%
4630   \ifglssentryexists{#1}%
4631   {%
4632     \global\letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
4633     \global\letcs{\@gls@location}{glo\@glsdetoklabel{#1}@location}%
4634     \ifglshasparent{#1}%
4635     {%
4636       \gls@level=\csuse{glo\@glsdetoklabel{#1}@level}\relax
4637       \ifdefvoid{\@gls@location}%

```



```

4638     {%
4639         \ifdefvoid{\@gls@loclist}%
4640     {%
4641         \subglossentry{\gls@level}{#1}{}%
4642     }%
4643     {%
4644         \subglossentry{\gls@level}{#1}%
4645         {%
4646             \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4647         }%
4648     }%
4649 }%
4650 {%
4651     \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
4652 }%
4653 }%
4654 {%

4655     \ifdefvoid{\@gls@location}%
4656     {%
4657         \ifdefvoid{\@gls@loclist}
4658         {%
4659             \glossentry{#1}{}%
4660         }%
4661         {%
4662             \glossentry{#1}%
4663             {%
4664                 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4665             }%
4666         }%
4667     }%
4668     {%
4669         \glossentry{#1}%
4670         {%
4671             \glossaryentrynumbers{\@gls@location}%
4672         }%
4673     }%
4674 }%
4675 }%
4676 {}%
4677 }

```

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`

```
4678 \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>}`

```

4679 \newcommand*{\@glsxtrnewgls}[4]{%
4680   \ifdef{#3}%
4681   {%
4682     \PackageError{glossaries-extra}{Command \string#3\space already
4683 defined}{}%
4684   }%
4685   {%
4686     \ifcsdef{@#4like@#2}%
4687     {%
4688       \advance\@glsxtrnewgls@inner by \@ne
4689       \def\@glsxtrnewgls@innercsname{@#4like\number\@glsxtrnewgls@inner @#2}%
4690     }%
4691     {\def\@glsxtrnewgls@innercsname{@#4like@#2}}%
4692     \expandafter\newrobustcmd\expandafter*\expandafter
4693       #3\expandafter{\expandafter\@gls@hyp@opt\csname\@glsxtrnewgls@innercsname\endcsname}%
4694     \ifstrempy{#1}%
4695     {%
4696       \expandafter\newcommand\expandafter*\csname\@glsxtrnewgls@innercsname\endcsname[2][ ]{%
4697         \new@ifnextchar[%
4698           {\csname @#4@\endcsname{##1}{#2##2}}%
4699           {\csname @#4@\endcsname{##1}{#2##2} [ ]}%
4700         }%
4701       }%
4702     }%
4703     \expandafter\newcommand\expandafter*\csname\@glsxtrnewgls@innercsname\endcsname[2][ ]{%
4704       \new@ifnextchar[%
4705         {\csname @#4@\endcsname{#1,##1}{#2##2}}%
4706         {\csname @#4@\endcsname{#1,##1}{#2##2} [ ]}%
4707       }%
4708     }%
4709   }%
4710 }
```

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

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

```

4711 \newrobustcmd*{\glsxtrnewgls}[3][ ]{%
4712   \@glsxtrnewgls{#1}{#2}{#3}{gls}%
4713 }
```

`\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.

```
4714 \newrobustcmd*{\lsxtrnewglslike}[6] [] {%
4715   \@glsxtrnewgls{#1}{#2}{#3}{gls}%
4716   \@glsxtrnewgls{#1}{#2}{#4}{glspl}%
4717   \@glsxtrnewgls{#1}{#2}{#5}{Gls}%
4718   \@glsxtrnewgls{#1}{#2}{#6}{Glspl}%
4719 }
```

`\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.

```
4720 \newrobustcmd*{\lsxtrnewGLSlike}[4] [] {%
4721   \@glsxtrnewgls{#1}{#2}{#3}{GLS}%
4722   \@glsxtrnewgls{#1}{#2}{#4}{GLSpl}%
4723 }
```

`\glsxtrnewrgls` As `\glsxtrnewgls` but for `\rgls`.

```
4724 \newrobustcmd*{\glsxtrnewrgls}[3] [] {%
4725   \@glsxtrnewgls{#1}{#2}{#3}{rgls}%
4726 }
```

`\sxtrnewrglslike` As `\glsxtrnewglslike` but for `\rgls` etc.

```
4727 \newrobustcmd*{\sxtrnewrglslike}[6] [] {%
4728   \@glsxtrnewgls{#1}{#2}{#3}{rgls}%
4729   \@glsxtrnewgls{#1}{#2}{#4}{rglspl}%
4730   \@glsxtrnewgls{#1}{#2}{#5}{rGls}%
4731   \@glsxtrnewgls{#1}{#2}{#6}{rGlspl}%
4732 }
```

`\sxtrnewrGLSlike` As `\glsxtrnewGLSlike` but for `\rGLS` etc.

```
4733 \newrobustcmd*{\sxtrnewrGLSlike}[4] [] {%
4734   \@glsxtrnewgls{#1}{#2}{#3}{rGLS}%
4735   \@glsxtrnewgls{#1}{#2}{#4}{rGLSpl}%
4736 }
```

Provide easy access to record count fields.

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

```
4737 \newcommand*{\GlsXtrTotalRecordCount}[1] {%
4738   \ifcsdef{glo@glstoklabel{#1}@recordcount}%
4739   {\csname glo@glstoklabel{#1}@recordcount\endcsname}%
4740   {0}%
4741 }
```

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

```
4742 \newcommand*{\GlsXtrRecordCount}[2] {%
```

```

4743 \ifcsdef{glo@\glxstrdetoklabel{#1}@recordcount.#2}%
4744 {\csname glo@\glxstrdetoklabel{#1}@recordcount.#2\endcsname}%
4745 {0}%
4746 }

```

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.

```

4747 \newcommand*{\GlsXtrLocationRecordCount}[3]{%
4748 \ifcsdef{glo@\glxstrdetoklabel{#1}@recordcount.#2.\glxstrdetoklocation{#3}}%
4749 {\csname glo@\glxstrdetoklabel{#1}@recordcount.#2.\glxstrdetoklocation{#3}\endcsname}%
4750 {0}%
4751 }

```

trdetoklocation

```

4752 \newcommand*{\glxstrdetoklocation}[1]{#1}

```

ablerecordcount

```

4753 \newcommand*{\glxstreablerecordcount}{%
4754 \renewcommand*{\gls}{\rgls}%
4755 \renewcommand*{\Gls}{\rGls}%
4756 \renewcommand*{\glspl}{\rglspl}%
4757 \renewcommand*{\Glspl}{\rGlspl}%
4758 \renewcommand*{\GLS}{\rGLS}%
4759 \renewcommand*{\GLSpl}{\rGLSpl}%
4760 }

```

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

```

4761 \newcommand*{\glxstrrecordtriggervalue}[1]{%
4762 \GlsXtrTotalRecordCount{#1}%
4763 }

```

dCountAttribute

```

4764 \newcommand*{\GlsXtrSetRecordCountAttribute}[2]{%
4765 \@for\@glxstr@cat:=#1\do
4766 {%
4767 \ifdefempty{\@glxstr@cat}{}%
4768 {%
4769 \glsssetcategoryattribute{\@glxstr@cat}{recordcount}{#2}%
4770 }%
4771 }%
4772 }

```

rifrecordtrigger

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

```

4773 \newcommand*{\glxtrifrecordtrigger}[3]{%
4774   \glshasattribute{#1}{recordcount}%
4775   {%
4776     \ifnum\glxtrrecordtriggervalue{#1}>\glsggetattribute{#1}{recordcount}\relax
4777     #3%
4778     \else
4779     #2%
4780     \fi
4781   }%
4782   {#3}%
4783 }

```

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

```

4784 \newcommand*{\@glxtr@rglstrigger@record}[3]{%
4785   \edef\glslabel{\glsdetoklabel{#2}}%
4786   \let\@gls@link@label\glslabel
4787   \def\@glxtr@thevalue{%
4788     \def\@glxtr@theHvalue{\@glxtr@thevalue}%
4789     \def\@glsnumberformat{glstriggerrecordformat}%
4790     \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
4791     \edef\glstype{\csname glo@\glslabel @type\endcsname}%
4792     \def\@glxtr@thevalue{%
4793       \def\@glxtr@theHvalue{\@glxtr@thevalue}%
4794       \glxtrinitwrgloss
4795       \glslinkpresetkeys
4796       \setkeys{glslink}{#1}%
4797       \glslinkpostsetkeys
4798       \ifdefempty{\@glxtr@thevalue}%
4799       {%
4800         \@gls@saveentrycounter
4801       }%
4802       {%
4803         \let\thegl Sentrycounter\@glxtr@thevalue
4804         \def\theHgl Sentrycounter{\@glxtr@theHvalue}%
4805       }%
4806       \ifglxtrinitwrglossbefore
4807         \@do@wrglossary{#2}%
4808       \fi
4809       #3%
4810       \ifglxtrinitwrglossbefore
4811       \else
4812         \@do@wrglossary{#2}%
4813       \fi
4814       \ifKV@glslink@local
4815         \glslocalunset{#2}%
4816       \else
4817         \glsunset{#2}%
4818       \fi

```

```

4819 }

gerrecordformat Typically won't be used as it should be recognised as a special type of ignored location by
bib2gls.
4820 \newcommand*{\glstriggerrecordformat}[1]{

\rgls
4821 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}

\@rgls
4822 \newcommand*{\@rgls}[2][]{%
4823   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}%
4824 }

\@rgls@
4825 \def\@rgls@#1#2[#3]{%
4826   \glstrifrecordtrigger{#2}%
4827   {%
4828     \@glstr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
4829   }%
4830   {%
4831     \@gls@{#1}{#2}[#3]%
4832   }%
4833 }%

\rglspl
4834 \newrobustcmd*{\rglspl}{\@gls@hyp@opt\@rglspl}

\@rglspl
4835 \newcommand*{\@rglspl}[2][]{%
4836   \new@ifnextchar[{\@rglspl@{#1}{#2}}{\@rglspl@{#1}{#2}[]}%
4837 }

\@rglspl@
4838 \def\@rglspl@#1#2[#3]{%
4839   \glstrifrecordtrigger{#2}%
4840   {%
4841     \@glstr@rglstrigger@record{#1}{#2}{\rglsplformat{#2}{#3}}%
4842   }%
4843   {%
4844     \@glspl@{#1}{#2}[#3]%
4845   }%
4846 }%

\rGls
4847 \newrobustcmd*{\rGls}{\@gls@hyp@opt\@rGls}

```

```

\@rGls
4848 \newcommand*{\@rGls}[2][{}]{%
4849   \new@ifnextchar[{\@rGls@{#1}{#2}}{\@rGls@{#1}{#2}[]}%
4850 }

\@rGls@
4851 \def\@rGls@#1#2[#3]{%
4852   \glxtrifrecordtrigger{#2}%
4853   {%
4854     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsformat{#2}{#3}}%
4855   }%
4856   {%
4857     \@Gls@{#1}{#2}[#3]%
4858   }%
4859 }%

\rGlspl
4860 \newrobustcmd*{\rGlspl}{\@glshyp@opt\@rGlspl}

\@rGlspl
4861 \newcommand*{\@rGlspl}[2][{}]{%
4862   \new@ifnextchar[{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2}[]}%
4863 }

\@rGlspl@
4864 \def\@rGlspl@#1#2[#3]{%
4865   \glxtrifrecordtrigger{#2}%
4866   {%
4867     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
4868   }%
4869   {%
4870     \@Glspl@{#1}{#2}[#3]%
4871   }%
4872 }%

\rGLS
4873 \newrobustcmd*{\rGLS}{\@glshyp@opt\@rGLS}

\@rGLS
4874 \newcommand*{\@rGLS}[2][{}]{%
4875   \new@ifnextchar[{\@rGLS@{#1}{#2}}{\@rGLS@{#1}{#2}[]}%
4876 }

\@rGLS@
4877 \def\@rGLS@#1#2[#3]{%
4878   \glxtrifrecordtrigger{#2}%
4879   {%
4880     \@glxtr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%

```

```

4881 }%
4882 {%
4883   \@GLS@{#1}{#2}[#3]%
4884 }%
4885 }%

```

\rGLSp1

```
4886 \newrobustcmd*{\rGLSp1}{\@gls@hyp@opt\rGLSp1}
```

\@rGLSp1

```

4887 \newcommand*{\@rGLSp1}[2][{}]{%
4888   \new@ifnextchar[{\@rGLSp1@{#1}{#2}}{\@rGLSp1@{#1}{#2}[{}]}%
4889 }

```

\@rGLSp1@

```

4890 \def\rGLSp1@#1#2[#3]{%
4891   \glsxtrifrecordtrigger{#2}%
4892   {%
4893     \@glsxtr@rglstrigger@record{#1}{#2}{\rGLSp1format{#2}{#3}}%
4894   }%
4895   {%
4896     \@GLSp1@{#1}{#2}[#3]%
4897   }%
4898 }%

```

\rglsformat

```

4899 \newcommand*{\rglsformat}[2]{%
4900   \glsifregular{#1}
4901   {\glsentryfirst{#1}}%
4902   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
4903 }

```

\rglsp1format

```

4904 \newcommand*{\rglsp1format}[2]{%
4905   \glsifregular{#1}
4906   {\glsentryfirstplural{#1}}%
4907   {\ifglshaslong{#1}{\glsentrylongplural{#1}}{\glsentryfirstplural{#1}}}%#2%
4908 }

```

\rGlsformat

```

4909 \newcommand*{\rGlsformat}[2]{%
4910   \glsifregular{#1}
4911   {\Glsentryfirst{#1}}%
4912   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
4913 }

```

\rGlsplformat

```
4914 \newcommand*{\rGlsplformat}[2]{%
```



```

4915 \glsifregular{#1}
4916 {\Glsentryfirstplural{#1}}%
4917 {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}}%#2%
4918 }

```

\rGLSformat

```

4919 \newcommand*{\rGLSformat}[2]{%
4920 \expandafter\mfirstucMakeUppercase\expandafter{\rGLSformat{#1}{#2}}%
4921 }

```

\rGLSplformat

```

4922 \newcommand*{\rGLSplformat}[2]{%
4923 \expandafter\mfirstucMakeUppercase\expandafter{\rGLSplformat{#1}{#2}}%
4924 }

```

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.

```

4925 \newcommand{\@glsxtr@do@inc@linkcount}{%
  Does this entry have the linkcount attribute set?
4926 \glsifattribute{\glslabel}{linkcount}{true}%
4927 {%
  Does the counter exist?
4928 \ifcsdef{c@glxtr@linkcount@\glslabel}{}%
4929 {%
  Counter doesn’t exist, so define it.
4930 \newcounter{glxtr@linkcount@\glslabel}%
  If linkcountmaster is set, add to counter reset.
4931 \glshasattribute{\glslabel}{linkcountmaster}%
4932 {%
  Need to ensure values are fully expanded.
4933 \begingroup
4934 \edef\x{\endgroup\noexpand\@addtoreset{glxtr@linkcount@\glslabel}%
4935 {\glsggetattribute{\glslabel}{linkcountmaster}}}%
4936 \x
4937 }%

```

```

4938     {}%
4939   }%

  Increment counter:
4940   \glxstrinclinkcounter{glxstr@linkcount@glslabel}%
4941 }%
4942 {}%
4943 }

rlinkcounter  May be redefined to use \refstepcounter if required.
4944 \newcommand*{\glxstrinclinkcounter}[1]{\stepcounter{#1}}

linkCounterValue  Expands to the associated link counter register or 0 if not defined.
4945 \newcommand*{\GlsXtrLinkCounterValue}[1]{%
4946   \ifcsundef{c@glxstr@linkcount@#1}{0}{\csname c@glxstr@linkcount@#1\endcsname}%
4947 }

rTheLinkCounter  Expands to the display value of the associated link counter or 0 if not defined.
4948 \newcommand*{\GlsXtrTheLinkCounter}[1]{%
4949   \ifcsundef{theglsxtr@linkcount@#1}{0}%
4950   {\csname theglxstr@linkcount@#1\endcsname}%
4951 }

fLinkCounterDef  Tests if the counter has been defined
4952 \newcommand*{\GlsXtrIfLinkCounterDef}[3]{%
4953   \ifcsundef{theglsxtr@linkcount@#1}{#3}{#2}%
4954 }

LinkCounterName  Expands to the associated link counter name. (No check for existence.)
4955 \newcommand*{\GlsXtrLinkCounterName}[1]{glxstr@linkcount@#1}

ableLinkCounting  \GlsXtrEnableLinkCounting[master counter]{categories}

  Enable link counting for the given categories.
4956 \newcommand*{\GlsXtrEnableLinkCounting}[2][{}]{%
4957   \let\glxstr@inc@linkcount\@glxstr@do@inc@linkcount
4958   \@for\@glxstr@label:=#2\do
4959   {%
4960     \glsssetcategoryattribute{\@glxstr@label}{linkcount}{true}%
4961     \ifstrempy{#1}{}%
4962     {%
4963       \ifcsundef{c@#1}%
4964       {\@nocounterr{#1}}%
4965       {\glsssetcategoryattribute{\@glxstr@label}{linkcountmaster}{#1}}%
4966     }%
4967   }%
4968 }
4969 \@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.

```
4970 \@ifpackageloaded{glossaries-accsupp}
4971 {
```

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

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

```
4972 \newcommand*{\glsaccessname}[1]{%
4973 \glsnameaccessdisplay
4974 {%
4975 \glsentryname{#1}%
4976 }%
4977 {#1}%
4978 }
```

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

```
4979 \newcommand*{\Glsaccessname}[1]{%
4980 \glsnameaccessdisplay
4981 {%
4982 \Glsentryname{#1}%
4983 }%
4984 {#1}%
4985 }
```

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

```
4986 \newcommand*{\GLSaccessname}[1]{%
4987 \glsnameaccessdisplay
4988 {%
4989 \mfirstucMakeUppercase{\glsentryname{#1}}%
4990 }%
4991 {#1}%
4992 }
```

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

```
4993 \newcommand*{\glsaccesstext}[1]{%
4994 \glstextaccessdisplay
4995 {%
4996 \glsentrytext{#1}%
4997 }%
4998 {#1}%
4999 }
```

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

```
5000 \newcommand*{\Glsaccesstext}[1]{%
5001 \glstextaccessdisplay
5002 {%
5003 \Glsentrytext{#1}%
5004 }%
5005 {#1}%
5006 }
```

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

```
5007 \newcommand*{\GLSaccesstext}[1]{%
5008 \glstextaccessdisplay
5009 {%
5010 \mfirstucMakeUppercase{\Glsentrytext{#1}}%
5011 }%
5012 {#1}%
5013 }
```

`glsaccessplural` Display the plural value (no link and no check for existence).

```
5014 \newcommand*{\glsaccessplural}[1]{%
5015 \glspluralaccessdisplay
5016 {%
5017 \Glsentryplural{#1}%
5018 }%
5019 {#1}%
5020 }
```

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

```
5021 \newcommand*{\GLSaccessplural}[1]{%
5022 \glspluralaccessdisplay
5023 {%
5024 \Glsentryplural{#1}%
5025 }%
5026 {#1}%
5027 }
```

`GLSaccessplural` Display the plural value (no link and no check for existence) converted to upper case.

```
5028 \newcommand*{\GLSaccessplural}[1]{%
5029 \glspluralaccessdisplay
5030 {%
5031 \mfirstucMakeUppercase{\Glsentryplural{#1}}%
5032 }%
5033 {#1}%
5034 }
```

`\glsaccessfirst` Display the first value (no link and no check for existence).

```

5035 \newcommand*{\glsaccessfirst}[1]{%
5036   \glsfirstaccessdisplay
5037   {%
5038     \glstryfirst{#1}%
5039   }%
5040   {#1}%
5041 }

```

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

```

5042 \newcommand*{\Glsaccessfirst}[1]{%
5043   \glsfirstaccessdisplay
5044   {%
5045     \Glsstryfirst{#1}%
5046   }%
5047   {#1}%
5048 }

```

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

```

5049 \newcommand*{\GLSaccessfirst}[1]{%
5050   \glsfirstaccessdisplay
5051   {%
5052     \mfirstucMakeUppercase{\glstryfirst{#1}}%
5053   }%
5054   {#1}%
5055 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence).

```

5056 \newcommand*{\glsaccessfirstplural}[1]{%
5057   \glsfirstpluralaccessdisplay
5058   {%
5059     \glstryfirstplural{#1}%
5060   }%
5061   {#1}%
5062 }

```

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

```

5063 \newcommand*{\Glsaccessfirstplural}[1]{%
5064   \glsfirstpluralaccessdisplay
5065   {%
5066     \Glsstryfirstplural{#1}%
5067   }%
5068   {#1}%
5069 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence) converted to upper case.

```

5070 \newcommand*{\GLSaccessfirstplural}[1]{%

```

```

5071 \glsfirstpluralaccessdisplay
5072 {%
5073     \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
5074 }%
5075 {#1}%
5076 }

```

glsaccesssymbol Display the symbol value (no link and no check for existence).

```

5077 \newcommand*{\glsaccesssymbol}[1]{%
5078     \glssymbolaccessdisplay
5079     {%
5080         \glsentrysymbol{#1}%
5081     }%
5082     {#1}%
5083 }

```

Glsaccesssymbol Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```

5084 \newcommand*{\Glsaccesssymbol}[1]{%
5085     \glssymbolaccessdisplay
5086     {%
5087         \glsentrysymbol{#1}%
5088     }%
5089     {#1}%
5090 }

```

GLSaccesssymbol Display the symbol value (no link and no check for existence) converted to upper case.

```

5091 \newcommand*{\GLSaccesssymbol}[1]{%
5092     \glssymbolaccessdisplay
5093     {%
5094         \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
5095     }%
5096     {#1}%
5097 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence).

```

5098 \newcommand*{\glsaccesssymbolplural}[1]{%
5099     \glssymbolpluralaccessdisplay
5100     {%
5101         \glsentrysymbolplural{#1}%
5102     }%
5103     {#1}%
5104 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

5105 \newcommand*{\Glsaccesssymbolplural}[1]{%
5106     \glssymbolpluralaccessdisplay

```

```

5107    {%
5108        \Glsentrysymbolplural{#1}%
5109    }%
5110    {#1}%
5111 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence) converted to upper case.

```

5112 \newcommand*{\GLSaccesssymbolplural}[1]{%
5113     \glssymbolpluralaccessdisplay
5114     {%
5115         \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
5116     }%
5117     {#1}%
5118 }

```

\glsaccessdesc Display the desc value (no link and no check for existence).

```

5119 \newcommand*{\glsaccessdesc}[1]{%
5120     \glsdescriptionaccessdisplay
5121     {%
5122         \glentrydesc{#1}%
5123     }%
5124     {#1}%
5125 }

```

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

```

5126 \newcommand*{\Glsaccessdesc}[1]{%
5127     \glsdescriptionaccessdisplay
5128     {%
5129         \Glsentrydesc{#1}%
5130     }%
5131     {#1}%
5132 }

```

\GLSaccessdesc Display the desc value (no link and no check for existence) converted to upper case.

```

5133 \newcommand*{\GLSaccessdesc}[1]{%
5134     \glsdescriptionaccessdisplay
5135     {%
5136         \mfirstucMakeUppercase{\glentrydesc{#1}}%
5137     }%
5138     {#1}%
5139 }

```

ccessdescplural Display the descplural value (no link and no check for existence).

```

5140 \newcommand*{\glsaccessdescplural}[1]{%
5141     \glsdescriptionpluralaccessdisplay
5142     {%
5143         \glentrydescplural{#1}%

```

```

5144     }%
5145     {#1}%
5146 }

```

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

```

5147 \newcommand*{\Glsaccessdescplural}[1]{%
5148   \glsdescriptionpluralaccessdisplay
5149   {%
5150     \Glsentrydescplural{#1}%
5151   }%
5152   {#1}%
5153 }

```

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

```

5154 \newcommand*{\GLSaccessdescplural}[1]{%
5155   \glsdescriptionpluralaccessdisplay
5156   {%
5157     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
5158   }%
5159   {#1}%
5160 }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

5161 \newcommand*{\glsaccessshort}[1]{%
5162   \glsshortaccessdisplay
5163   {%
5164     \glsentryshort{#1}%
5165   }%
5166   {#1}%
5167 }

```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

5168 \newcommand*{\Glsaccessshort}[1]{%
5169   \glsshortaccessdisplay
5170   {%
5171     \Glsentryshort{#1}%
5172   }%
5173   {#1}%
5174 }

```

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

```

5175 \newcommand*{\GLSAccessshort}[1]{%
5176   \glsshortaccessdisplay
5177   {%
5178     \mfirstucMakeUppercase{\glsentryshort{#1}}%
5179   }%

```



```

5180     {#1}%
5181 }

```

lsaccessshortpl Display the short plural form (no link and no check for existence).

```

5182 \newcommand*{\lsaccessshortpl}[1]{%
5183   \glshortpluralaccessdisplay
5184   {%
5185     \glentryshortpl{#1}%
5186   }%
5187   {#1}%
5188 }

```

LSaccessshortpl Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

5189 \newcommand*{\LSaccessshortpl}[1]{%
5190   \glshortpluralaccessdisplay
5191   {%
5192     \GLentryshortpl{#1}%
5193   }%
5194   {#1}%
5195 }

```

LSaccessshortpl Display the shortplural value (no link and no check for existence) converted to upper case.

```

5196 \newcommand*{\LSaccessshortpl}[1]{%
5197   \glshortpluralaccessdisplay
5198   {%
5199     \mfirstucMakeUppercase{\glentryshortpl{#1}}%
5200   }%
5201   {#1}%
5202 }

```

\glsaccesslong Display the long form (no link and no check for existence).

```

5203 \newcommand*{\glsaccesslong}[1]{%
5204   \glslongaccessdisplay{\glentrylong{#1}}{#1}%
5205 }

```

\Glsaccesslong Display the long form (no link and no check for existence).

```

5206
5207 \newcommand*{\Glsaccesslong}[1]{%
5208   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
5209 }

```

\GLSaccesslong Display the long value (no link and no check for existence) converted to upper case.

```

5210 \newcommand*{\GLSaccesslong}[1]{%
5211   \glslongaccessdisplay
5212   {%
5213     \mfirstucMakeUppercase{\glentrylong{#1}}%
5214   }%

```

```

5215     {#1}%
5216 }

```

`\glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5217 \newcommand*{\glsaccesslongpl}[1]{%
5218     \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
5219 }

```

`\Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5220
5221 \newcommand*{\Glsaccesslongpl}[1]{%
5222     \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
5223 }

```

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

```

5224 \newcommand*{\GLSaccesslongpl}[1]{%
5225     \glslongpluralaccessdisplay
5226     {%
5227         \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
5228     }%
5229     {#1}%
5230 }

```

End of if part

```

5231 }
5232 {

```

No accessibility support. Just define these commands to do `\glsentry<xxx>`

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

```

5233 \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.

```

5234 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

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

```

5235 \newcommand*{\GLSaccessname}[1]{%
5236     \protect\mfirstucMakeUppercase{\glsentryname{#1}}}

```

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

```

5237 \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.

```

5238 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}

```

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

```

5239 \newcommand*{\GLSaccessstext}[1]{%
5240 \protect\mfirstucMakeUppercase{\glstentrytext{#1}}}
```

`glsaccessplural` Display the plural value (no link and no check for existence).

```

5241 \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.

```

5242 \newcommand*{\Glsaccessplural}[1]{\Glstentryplural{#1}}
```

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.

```

5243 \newcommand*{\GLSaccessplural}[1]{%
5244 \protect\mfirstucMakeUppercase{\glstentryplural{#1}}}
```

`\glsaccessfirst` Display the first value (no link and no check for existence).

```

5245 \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.

```

5246 \newcommand*{\Glsaccessfirst}[1]{\Glstentryfirst{#1}}
```

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

```

5247 \newcommand*{\GLSaccessfirst}[1]{%
5248 \protect\mfirstucMakeUppercase{\glstentryfirst{#1}}}
```

`cessfirstplural` Display the firstplural value (no link and no check for existence).

```

5249 \newcommand*{\glsaccessfirstplural}[1]{\glstentryfirstplural{#1}}
```

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

```

5250 \newcommand*{\Glsaccessfirstplural}[1]{\Glstentryfirstplural{#1}}
```

`GLSaccessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.

```

5251 \newcommand*{\GLSaccessfirstplural}[1]{%
5252 \protect\mfirstucMakeUppercase{\glstentryfirstplural{#1}}}
```

`glsaccesssymbol` Display the symbol value (no link and no check for existence).

```

5253 \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.

```

5254 \newcommand*{\Glsaccesssymbol}[1]{\Glstentrysymbol{#1}}
```

`GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.

```

5255 \newcommand*{\GLSaccesssymbol}[1]{%
5256 \protect\mfirstucMakeUppercase{\glstentrysymbol{#1}}}
```

esssymbolplural Display the symbolplural value (no link and no check for existence).
5257 \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.
5258 \newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}

esssymbolplural Display the symbolplural value (no link and no check for existence). converted to upper case.
5259 \newcommand*{\GLSaccesssymbolplural}[1]{%
5260 \protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}

\glsaccessdesc Display the desc value (no link and no check for existence).
5261 \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.
5262 \newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}

\GLSaccessdesc Display the desc value (no link and no check for existence). converted to upper case.
5263 \newcommand*{\GLSaccessdesc}[1]{%
5264 \protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}

ccessdescplural Display the descplural value (no link and no check for existence).
5265 \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.
5266 \newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}

ccessdescplural Display the descplural value (no link and no check for existence). converted to upper case.
5267 \newcommand*{\GLSaccessdescplural}[1]{%
5268 \protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}

\glsaccessshort Display the short form (no link and no check for existence).
5269 \newcommand*{\glsaccessshort}[1]{\glsentryshort{#1}}

\Glsaccessshort Display the short form with first letter converted to uppercase (no link and no check for existence).
5270 \newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}

\GLSaccessshort Display the short value (no link and no check for existence). converted to upper case.
5271 \newcommand*{\GLSaccessshort}[1]{%
5272 \protect\mfirstucMakeUppercase{\glsentryshort{#1}}}

lsaccessshortpl Display the short plural form (no link and no check for existence).
5273 \newcommand*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}

`\lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```
5274 \newcommand*{\GLsaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

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

```
5275 \newcommand*{\GLSaccessshortpl}[1]{%
```

```
5276 \protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}
```

`\glsaccesslong` Display the long form (no link and no check for existence).

```
5277 \newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}
```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```
5278 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

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

```
5279 \newcommand*{\GLSaccesslong}[1]{%
```

```
5280 \protect\mfirstucMakeUppercase{\glsentrylong{#1}}}
```

`\glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5281 \newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}
```

`\Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5282 \newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

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

```
5283 \newcommand*{\GLSaccesslongpl}[1]{%
```

```
5284 \protect\mfirstucMakeUppercase{\glsentrylongpl{#1}}}
```

End of else part

```
5285 }
```

1.6 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.

```
5286 \glsaddstoragekey{category}{general}{\glscategory}
```

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.

```
5287 \newcommand{\glsifcategory}[4]{%
```

```
5288 \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
```

```
5289 }
```

Categories can have attributes.

categoryattribute

```
\glissetcategoryattribute{<category>}{<attribute-label>}{<value>}
```

Set (or override if already set) an attribute for the given category.

```
5290 \newcommand*{\glissetcategoryattribute}[3]{%
5291   \csdef{@glsxtr@categoryattr@@#1@#2}{#3}%
5292 }
```

categoryattribute

```
\glsggetcategoryattribute{<category>}{<attribute-label>}
```

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
5293 \newcommand*{\glsggetcategoryattribute}[2]{%
5294   \csuse{@glsxtr@categoryattr@@#1@#2}%
5295 }
```

categoryattribute

```
\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}
```

Tests if the category has the given attribute set.

```
5296 \newcommand*{\glshascategoryattribute}[4]{%
5297   \ifcsvoid{@glsxtr@categoryattr@@#1@#2}{#4}{#3}%
5298 }
```

\glissetattribute

```
\glissetattribute{<entry label>}{<attribute-label>}{<value>}
```

Short cut where the category label is obtained from the entry information.

```
5299 \newcommand*{\glissetattribute}[3]{%
5300   \glissetcategoryattribute{\glscategory{#1}}{#2}{#3}%
5301 }
```

\glsggetattribute

```
\glsggetattribute{<entry label>}{<attribute-label>}
```

Short cut where the category label is obtained from the entry information.

```
5302 \newcommand*{\glsggetattribute}[2]{%
5303   \glsggetcategoryattribute{\glscategory{#1}}{#2}%
5304 }
```

\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.

```
5305 \newcommand*{\glshasattribute}[4]{%
5306   \ifglentryexists{#1}%
5307   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
5308   {#4}%
5309 }
```

categoryattribute

`\glcifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

True if category has the attribute with the given value.

```
5310 \newcommand{\glcifcategoryattribute}[5]{%
5311   \ifcsundef{@glxtr@categoryattr@#1@#2}%
5312   {#5}%
5313   {\ifcsstring{@glxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
5314 }
```

\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.

```
5315 \newcommand{\glcifattribute}[5]{%
5316   \ifglentryexists{#1}%
5317   {\glcifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
5318   {#5}%
5319 }
```

Set attributes for the default general category:

```
5320 \glsetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
5321 \glsetcategoryattribute{acronym}{regular}{true}
```

regularcategory

Convenient shortcut to create add the regular attribute.

```
5322 \newcommand*{\glsetregularcategory}[1]{%
5323   \glsetcategoryattribute{#1}{regular}{true}%
5324 }
```

`fregularcategory` `\glsifregularcategory{<category>}{<true part>}{<>false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```
5325 \newcommand{\glsifregularcategory}[3]{%
5326   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
5327 }
```

`tregularcategory` `\glsifnotregularcategory{<category>}{<true part>}{<>false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```
5328 \newcommand{\glsifnotregularcategory}[3]{%
5329   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
5330 }
```

`\glsifregular` `\glsifregular{<entry label>}{<true part>}{<>false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
5331 \newcommand{\glsifregular}[3]{%
5332   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
5333 }
```

`\glsifnotregular` `\glsifnotregular{<entry label>}{<true part>}{<>false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
5334 \newcommand{\glsifnotregular}[3]{%
5335   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
5336 }
```

`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.

```
5337 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
```



```

5338 \forallglossaries[#1]{#3}%
5339 {%
5340     \forallsentries[#3]{#4}%
5341     {%
5342         \glsifcategory{#4}{#2}{#5}{}%
5343     }%
5344 }%
5345 }

```

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.

```

5346 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
5347     \forallglossaries[#1]{#4}%
5348     {%
5349         \forallsentries[#4]{#5}%
5350         {%
5351             \glsifattribute{#5}{#2}{#3}{#6}{}%
5352         }%
5353     }%
5354 }

```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to index and add `\glstrpostdescription`.

```

5355 \ifdef\newterm
5356 {%

```

`\newterm`

```

5357 \renewcommand*{\newterm}[2][ ]{%
5358     \newglossaryentry{#2}%
5359     {type=index,category=index,name={#2},%
5360     description={\glstrpostdescription\nopostdesc},#1}%
5361 }

```

Indexed terms are regular by default.

```

5362 \glssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

5363 \newcommand*{\glstrpostdescindex}{}
5364 }
5365 {}

```

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.

```
5366 \ifdef\printsymbols
5367 {%
```

`glsxtrnewsymbol` Unlike `\newterm`, this has a separate argument for the label (since the symbol will likely contain commands).

```
5368 \newcommand*{\glsxtrnewsymbol}[3] [] {%
5369 \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
5370 }
```

Symbols are regular by default.

```
5371 \glsssetcategoryattribute{symbol}{regular}{true}
```

`rpostdescsymbol`

```
5372 \newcommand*{\glsxtrpostdescsymbol}{}
5373 }
5374 {}
```

Similar for the numbers option.

```
5375 \ifdef\printnumbers
5376 {%
```

`glsxtrnewnumber`

```
5377 \ifdef\printnumbers
5378 \newcommand*{\glsxtrnewnumber}[3] [] {%
5379 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
5380 }
```

Numbers are regular by default.

```
5381 \glsssetcategoryattribute{number}{regular}{true}
```

`rpostdescnumber`

```
5382 \newcommand*{\glsxtrpostdescnumber}{}
5383 }
5384 {}
```

`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.

```
5385 \newcommand*{\glsxtrsetcategory}[2] {%
5386 \@for\@glsxtr@label:=#1\do
5387 {%
5388 \glsfieldxddef{\@glsxtr@label}{category}{#2}%
5389 }%
5390 }
```

`\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.

```

5391 \newcommand*{\glxtrsetcategoryforall}[2]{%
5392   \forallglossaries[#1]{\@glxtr@type}{%
5393     \forallglsentries[\@glxtr@type]{\@glxtr@label}%
5394     {%
5395       \glsfieldxdef{\@glxtr@label}{category}{#2}%
5396     }%
5397   }%
5398 }
```

`\glxtrfieldtitlecase` `\glxtrfieldtitlecase{\langle label \rangle}{\langle field \rangle}`

Apply title casing to the contents of the given field.

```

5399 \newcommand*{\glxtrfieldtitlecase}[2]{%
5400   \expandafter\glxtrfieldtitlecasesecs\expandafter
5401   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
5402 }
```

`\glxtrfieldtitlecasesecs` The command used by `\glxtrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```

5403 \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.

```

5404 \ifpackageloaded{glossaries-accsupp}
5405 {
5406   \renewcommand*{\glossentrydesc}[1]{%
5407     \glsdoifexistsorwarn{#1}%
5408     {%
5409       \glssetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the `glossdescfont` attribute to determine the font applied.

```

5410     \glshasattribute{#1}{glossdescfont}%
5411     {%
5412       \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5413       \ifcsdef{\@glxtr@attrval}%
5414       {%
5415         \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5416       }%
5417     }%
5418     \GlossariesExtraWarning{Unknown control sequence name
5419     ‘\@glxtr@attrval’ supplied in glossdescfont attribute
```

```

5420         for entry '#1'. Ignoring}%
5421         \let\@glxtr@glossdescfont\@firstofone
5422     }%
5423 }%
5424 {\let\@glxtr@glossdescfont\@firstofone}%
5425 \glsifattribute{#1}{glossdesc}{firstuc}%
5426 {%
5427     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5428 }%
5429 {%
5430     \glsifattribute{#1}{glossdesc}{title}%
5431     {%
5432         \@glxtr@do@titlecaps@warn
5433         \glsdescriptionaccessdisplay
5434         {%
5435             \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
5436         }%
5437         {#1}%
5438     }%
5439     {%
5440         \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5441     }%
5442 }%
5443 }%
5444 }
5445 }
5446 {
5447 \renewcommand*{\glossentrydesc}[1]{%
5448     \glsdoifexistsorwarn{#1}%
5449     {%
5450         \glssetabbrvfmt{\glscategory{#1}}%
5451         \glsattribute{#1}{glossdescfont}%
5452         {%
5453             \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5454             \ifcsdef{\@glxtr@attrval}%
5455             {%
5456                 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5457             }%
5458             {%
5459                 \GlossariesExtraWarning{Unknown control sequence name
5460                     '\@glxtr@attrval' supplied in glossdescfont attribute
5461                     for entry '#1'. Ignoring}%
5462                 \let\@glxtr@glossdescfont\@firstofone
5463             }%
5464         }%
5465         {\let\@glxtr@glossdescfont\@firstofone}%
5466         \glsifattribute{#1}{glossdesc}{firstuc}%
5467         {%
5468             \@glxtr@glossdescfont{\Glsentrydesc{#1}}%

```

```

5469     }%
5470     {%
5471         \glsifattribute{#1}{glossdesc}{title}%
5472         {%
5473             \@glsxtr@do@titlecaps@warn
5474             \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
5475         }%
5476         {%
5477             \@glsxtr@glossdescfont{\glsentrydesc{#1}}%
5478         }%
5479     }%
5480 }%
5481 }
5482 }

```

`\glossentryname` If the `glossname` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

5483 \ifpackageloaded{glossaries-accsupp}
5484 {
5485   \renewcommand*{\glossentryname}[1]{%
5486     \@glsdoifexistsorwarn{#1}%
5487     {%
5488       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```

5489   \glsifattribute{#1}{glossnamefont}%
5490   {%
5491     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5492     \ifcsdef{\@glsxtr@attrval}%
5493     {%
5494       \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5495     }%
5496     {%
5497       \GlossariesExtraWarning{Unknown control sequence name
5498         ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
5499         for entry ‘#1’. Reverting to default \string\glsnamefont}%
5500       \let\@glsxtr@glossnamefont\glsnamefont
5501     }%
5502   }%
5503   {\let\@glsxtr@glossnamefont\glsnamefont}%
5504   \glsifattribute{#1}{glossname}{firstuc}%
5505   {%
5506     \glsnameaccessdisplay
5507     {%
5508       \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5509     }%
5510     {#1}%
5511   }%
5512   {%
5513     \glsifattribute{#1}{glossname}{title}%

```

```

5514      {%
5515      \@glstr@do@titlecaps@warn
5516      \glsnameaccessdisplay
5517      {%
5518      \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}}%
5519      }%
5520      {#1}%
5521      }%
5522      {%
5523      \glsifattribute{#1}{glossname}{uc}%
5524      {%
5525      \glsnameaccessdisplay
5526      {%

```

Hide the label from the upper-casing command.

```

5527      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5528      \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
5529      }%
5530      {#1}%
5531      }%
5532      {%
5533      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5534      \glsnameaccessdisplay
5535      {%
5536      \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
5537      }%
5538      {#1}%
5539      }%
5540      }%
5541      }%

```

Do post-name hook:

```

5542      \glstrpostnamehook{#1}%
5543      }%
5544  }
5545 }
5546 {
5547 \renewcommand*{\glossentryname}[1]{%
5548 \@glsdoifexistsorwarn{#1}%
5549 {%
5550 \glssetabbrvfmt{\glscategory{#1}}}%
5551 \glshasattribute{#1}{glossnamefont}%
5552 {%
5553 \edef\@glstr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5554 \ifcsdef{\@glstr@attrval}%
5555 {%
5556 \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5557 }%
5558 {%
5559 \GlossariesExtraWarning{Unknown control sequence name

```

```

5560         ‘\@glxtr@attrval’ supplied in glossnamefont attribute
5561         for entry ‘#1’. Reverting to default \string\glnamefont}%
5562         \let\@glxtr@glossnamefont\glnamefont
5563     }%
5564 }%
5565 {\let\@glxtr@glossnamefont\glnamefont}%
5566 \glsifattribute{#1}{glossname}{firstuc}%
5567 {%
5568     \@glxtr@glossnamefont{\Glsentryname{#1}}%
5569 }%
5570 {%
5571     \glsifattribute{#1}{glossname}{title}%
5572     {%
5573         \@glxtr@do@titlecaps@warn
5574         \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
5575     }%
5576     {%
5577         \glsifattribute{#1}{glossname}{uc}%
5578         {%

```

Hide the label from the upper-casing command.

```

5579         \letcs{\glo@name}{glo@glsetoklabel{#1}@name}%
5580         \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5581     }%
5582     {%

```

This little trick is used by glossaries to allow the user to redefine \glnamefont to use \makefirstuc. Support it even though they can now use the firstuc attribute.

```

5583         \letcs{\glo@name}{glo@glsetoklabel{#1}@name}%
5584         \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
5585     }%
5586 }%
5587 }%

```

Do post-name hook.

```

5588     \glxtrpostnamehook{#1}%
5589 }%
5590 }
5591 }

```

\Glossentryname Redefine to set the abbreviation format and accessibility support.

```

5592 \@ifpackageloaded{glossaries-accsupp}
5593 {
5594     \renewcommand*{\Glossentryname}[1]{%
5595         \@glsetoklabel{#1}%
5596     }%
5597     \glsetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

5598     \glshasattribute{#1}{glossnamefont}%
5599     {%

```

```

5600     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5601     \ifcsdef{\@glsxtr@attrval}%
5602     {%
5603         \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5604     }%
5605     {%
5606         \GlossariesExtraWarning{Unknown control sequence name
5607             '\@glsxtr@attrval' supplied in glossnamefont attribute
5608             for entry '#1'. Reverting to default \string\glsnamefont}%
5609         \let\@glsxtr@glossnamefont\glsnamefont
5610     }%
5611 }%
5612 {\let\@glsxtr@glossnamefont\glsnamefont}%
5613 \glsnameaccessdisplay
5614 {%
5615     \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5616 }%
5617 {#1}%

```

Do post-name hook:

```

5618     \glsxtrpostnamehook{#1}%
5619 }%
5620 }
5621 }
5622 {
5623     \renewcommand*{\Glossentryname}[1]{%
5624         \@glsdoifexistsorwarn{#1}%
5625         {%
5626             \glssetabbrvfmt{\glscategory{#1}}%
5627             \glsattribute{#1}{glossnamefont}%
5628             {%
5629                 \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5630                 \ifcsdef{\@glsxtr@attrval}%
5631                 {%
5632                     \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5633                 }%
5634                 {%
5635                     \GlossariesExtraWarning{Unknown control sequence name
5636                         '\@glsxtr@attrval' supplied in glossnamefont attribute
5637                         for entry '#1'. Reverting to default \string\glsnamefont}%
5638                     \let\@glsxtr@glossnamefont\glsnamefont
5639                 }%
5640             }%
5641             {\let\@glsxtr@glossnamefont\glsnamefont}%
5642             \@glsxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

5643     \glsxtrpostnamehook{#1}%
5644 }%
5645 }

```


5646 }

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.

```
5647 \newcommand*{\glxtrpostnamehook}[1]{%
5648   \let\@glsnumberformat\@glxtr@defaultnumberformat
5649   \glxtrdoautoindexname{#1}{indexname}%
```

Allow additional code regardless of category:

```
5650   \glsextrapostnamehook{#1}%
```

Allow categories to hook in here.

```
5651   \csuse{glxtrpostname\glscategory{#1}}%
5652 }
```

`trapostnamehook`

```
5653 \newcommand*{\glsextrapostnamehook}[1]{}%
```

`etaccessdisplay`

```
5654 \@ifpackageloaded{glossaries-accsupp}
5655 {
5656   \newcommand*{\glxtr@setaccessdisplay}[1]{%
5657     \ifcsdef{gls#1accessdisplay}%
5658     {\letcs\@glxtr@accessdisplay{gls#1accessdisplay}}%
5659     {%
```

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.

```
5660     \edef\@gls@thisval{#1}%
5661     \@for\@gls@map:=\@gls@keymap\do{%
5662       \edef\@this@key{\expandafter\@secondoftwo\@gls@map}%
5663       \ifdefequal{\@this@key}{\@gls@thisval}%
5664       {%
5665         \edef\@gls@thisval{\expandafter\@firstoftwo\@gls@map}%
5666         \@endfortrue
5667       }%
5668     }%
5669   }%
5670   \ifcsdef{gls\@gls@thisval accessdisplay}%
5671   {\letcs\@glxtr@accessdisplay{gls\@gls@thisval accessdisplay}}%
5672   {\let\@glxtr@accessdisplay\@firstoftwo}%
5673   }%
5674 }
5675 }
5676 {%
5677   \newcommand*{\glxtr@setaccessdisplay}[1]{%
```

```

5678 \let\@glxstr@accessdisplay\@firstoftwo}
5679 }

```

sentrynameother Provide a command that works like \glossentryname but accesses a different field (which must be supplied using its internal field label).

```

5680 \newrobustcmd*{\glossentrynameother}[2]{%
5681 \@glsoifexistsorwarn{#1}%
5682 {%

```

Accessibility support:

```

5683 \glxstr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

5684 \glsetabbrvfmt{\glscategory{#1}}%
5685 \glshasattribute{#1}{glossnamefont}%
5686 {%
5687 \edef\@glxstr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5688 \ifcsdef{\@glxstr@attrval}%
5689 {%
5690 \letcs{\@glxstr@glossnamefont}{\@glxstr@attrval}%
5691 }%
5692 {%
5693 \GlossariesExtraWarning{Unknown control sequence name
5694 '\@glxstr@attrval' supplied in glossnamefont attribute
5695 for entry '#1'. Reverting to default \string\glsgnamefont}%
5696 \let\@glxstr@glossnamefont\glsgnamefont
5697 }%
5698 }%
5699 {\let\@glxstr@glossnamefont\glsgnamefont}%
5700 \glsgifattribute{#1}{glossname}{firstuc}%
5701 {%
5702 \@glxstr@accessdisplay
5703 {\@glxstr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
5704 {#1}%
5705 }%
5706 {%
5707 \glsgifattribute{#1}{glossname}{title}%
5708 {%
5709 \@glxstr@do@titlecaps@warn
5710 \@glxstr@accessdisplay
5711 {\@glxstr@glossnamefont{\glxstr@field@titlecase{#1}{#2}}}%
5712 {#1}%
5713 }%
5714 {%
5715 \glsgifattribute{#1}{glossname}{uc}%
5716 {%
5717 \letcs{\glo@name}{\glo@\glsgdetoklabel{#1}{#2}}%
5718 \@glxstr@accessdisplay
5719 {\@glxstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
5720 {#1}%

```

```

5721     }%
5722     {%
5723         \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@#2}%
5724         \@glstr@accessdisplay
5725         {\expandafter\@glstr@glossnamefont\expandafter{\glo@name}}%
5726         {#1}%
5727     }%
5728 }%
5729 }%
    Do post-name hook.
5730     \glstrpostnamehook{#1}%
5731 }%
5732 }

```

`format@override` Determines if the format key should override the indexing attribute value.

```

5733 \newif\if@glstr@format@override
5734 \@glstr@format@overridefalse

```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

5735 \ifpackageloaded{hyperref}
5736 {
    If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
    don't add it.
5737     \ifHy@hyperindex
5738         \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5739             \@glstr@format@override true
5740             \appto\theindex{\let\glshypernumber\@firstofone}%
5741         }
5742     \else
5743         \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5744             \@glstr@format@override true
5745             \appto\theindex{\let\glshypernumber\hyperpage}%
5746         }
5747     \fi
5748 }
5749 {
5750     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5751         \@glstr@format@override true
5752     }
5753 }
5754 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

`doautoindexname`

```

5755 \newcommand*{\glstrdoautoindexname}[2]{%
5756     \glshasattribute{#1}{#2}%
5757     {%

```

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.

```
5758 \glsxtr@autoindex@setname{#1}%
```

If the attribute value is simply “true” don't add an encap, otherwise use the value as the encap.

```
5759 \protected@edef\glsxtr@attrval{\glsgetattribute{#1}{#2}}%
5760 \ifglsxtr@format@override

5761 \ifx\glsnumberformat\glsxtr@defaultnumberformat
5762 \else
5763 \let\glsxtr@attrval\glsnumberformat
5764 \fi
5765 \fi
5766 \ifdefstring{\glsxtr@attrval}{true}%
5767 {}%
5768 {\eappto\glo@name{\glsxtr@autoindex@encap\glsxtr@attrval}}%
5769 \expandafter\glsxtrautoindex\expandafter{\glo@name}%
5770 }%
5771 {}%
5772 }
```

glsxtrautoindex

```
5773 \newcommand*{\glsxtrautoindex}{\index}
```

toindex@setname Assign \glo@name for use with indexname attribute.

```
5774 \newcommand*{\glsxtr@autoindex@setname}[1]{%
5775 \protected@edef\glo@name{\glsxtrautoindexentry{#1}}%
5776 \glsxtrautoindexassignsort{\glo@sort}{#1}%
5777 \gls@checkmkidxchars\glo@sort
5778 \glsxtr@autoindex@doextra@esc\glo@sort
5779 \epreto\glo@name{\glo@sort\glsxtr@autoindex@at}%
5780 }
```

autoindexentry Command used for the actual part when auto-indexing.

```
5781 \newcommand*{\glsxtrautoindexentry}[1]{\string\glstryname{#1}}
```

indexassignsort Used to assign the sort value when auto-indexing.

```
5782 \newcommand*{\glsxtrautoindexassignsort}[2]{%
5783 \glsletentryfield{#1}{#2}{sort}%
5784 }
```

dex@doextra@esc

```
5785 \newcommand*{\glsxtr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
5786 \ifx\glsxtr@autoindex@esc\gls@quotechar
5787 \else
5788 \def\gls@checkedmkidx{}%
5789 \edef\@glsxtr@checkspch{%
```

```

5790 \noexpand\@glsxtr@autoindex@escquote\expandonce{#1}%
5791 \noexpand\@empty\@glsxtr@autoindex@esc\noexpand\@nnil
5792 \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5793 \@@glsxtr@checkspch
5794 \let#1\@gls@checkedmkidx\relax
5795 \fi

```

Escape actual character unless it has already been escaped.

```

5796 \ifx\@glsxtr@autoindex@at\@gls@actualchar
5797 \else
5798 \def\@gls@checkedmkidx{}%
5799 \edef\@@glsxtr@checkspch{%
5800 \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
5801 \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
5802 \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5803 \@@glsxtr@checkspch
5804 \let#1\@gls@checkedmkidx\relax
5805 \fi

```

Escape level character unless it has already been escaped.

```

5806 \ifx\@glsxtr@autoindex@level\@gls@levelchar
5807 \else
5808 \def\@gls@checkedmkidx{}%
5809 \edef\@@glsxtr@checkspch{%
5810 \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
5811 \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
5812 \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5813 \@@glsxtr@checkspch
5814 \let#1\@gls@checkedmkidx\relax
5815 \fi

```

Escape encap character unless it has already been escaped.

```

5816 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
5817 \else
5818 \def\@gls@checkedmkidx{}%
5819 \edef\@@glsxtr@checkspch{%
5820 \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
5821 \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
5822 \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5823 \@@glsxtr@checkspch
5824 \let#1\@gls@checkedmkidx\relax
5825 \fi
5826 }

```

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.

```

5827 \newcommand*{\@glsxtr@autoindex@at}{}

```

trSetActualChar Set the actual character.

```
5828 \newcommand*{\GlsXtrSetActualChar}[1]{%
5829   \gdef\@glstr@autoindex@at{#1}%
5830   \def\@glstr@autoindex@escat##1##2##3\@glstr@endescspch{%
5831     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escat}{##1}{##2}{##3}%
5832   }%
5833 }
5834 \@onlypreamble\GlsXtrSetActualChar
5835 \makeatother
5836 \GlsXtrSetActualChar{@}
5837 \makeatletter
```

autoindex@encap Encap character for use with \index.

```
5838 \newcommand*{\@glstr@autoindex@encap}{}
```

XtrSetEncapChar Set the encap character.

```
5839 \newcommand*{\GlsXtrSetEncapChar}[1]{%
5840   \gdef\@glstr@autoindex@encap{#1}%
5841   \def\@glstr@autoindex@escencap##1##2##3\@glstr@endescspch{%
5842     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escencap}{##1}{##2}{##3}%
5843   }%
5844 }
5845 \GlsXtrSetEncapChar{||}
5846 \@onlypreamble\GlsXtrSetEncapChar
```

autoindex@level Level character for use with \index.

```
5847 \newcommand*{\@glstr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
5848 \newcommand*{\GlsXtrSetLevelChar}[1]{%
5849   \gdef\@glstr@autoindex@level{#1}%
5850   \def\@glstr@autoindex@esclevel##1##2##3\@glstr@endescspch{%
5851     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@esclevel}{##1}{##2}{##3}%
5852   }%
5853 }
5854 \GlsXtrSetLevelChar{!}
5855 \@onlypreamble\GlsXtrSetLevelChar
```

r@autoindex@esc Escape character for use with \index.

```
5856 \newcommand*{\@glstr@autoindex@esc}{""}
```

lsXtrSetEscChar Set the escape character.

```
5857 \newcommand*{\GlsXtrSetEscChar}[1]{%
5858   \gdef\@glstr@autoindex@esc{#1}%
5859   \def\@glstr@autoindex@escquote##1##2##3\@glstr@endescspch{%
5860     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escquote}{##1}{##2}{##3}%
5861   }%
5862 }
```

```

5863 \GlsXtrSetEscChar{}
5864 \@onlypreamble\GlsXtrSetEscChar

```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```

5865 \ifdef\actualchar
5866 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
5867 {}

```

Quote character \quotechar:

```

5868 \ifdef\quotechar
5869 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
5870 {}

```

Level character \levelchar:

```

5871 \ifdef\levelchar
5872 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
5873 {}

```

Encap character \encapchar:

```

5874 \ifdef\encapchar
5875 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
5876 {}

```

leto@endescspch

```

5877 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}

```

toindex@esc@spch

```

\@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}

```

```

5878 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
5879   \@glstmpb=\expandafter{\@glstmpb\the\@glstmpb}%
5880   \toks@={#3}%
5881   \ifx\@nnil#3\relax
5882     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
5883   \else
5884     \ifx\@nnil#4\relax
5885       \edef\@glstmpb{\the\@glstmpb\the\@glstmpb}%
5886       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
5887         #4#5\@glxtr@endescspch}%
5888     \else
5889       \edef\@glstmpb{\the\@glstmpb\the\@glstmpb}
5890       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
5891         #4#5\@glxtr@endescspch}%
5892     \fi
5893   \fi
5894   \@glxtr@checkspch
5895 }

```

`\Glossentrydesc` Redefine to set the abbreviation format and accessibility support.

```
5896 \renewcommand*{\Glossentrydesc}[1]{%
5897   \glsdoifexistsorwarn{#1}%
5898   {%
5899     \glssetabbrvfmt{\glscategory{#1}}%
5900     \Glsaccessdesc{#1}%
5901   }%
5902 }
```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```
5903 \renewcommand*{\glossentrysymbol}[1]{%
5904   \glsdoifexistsorwarn{#1}%
5905   {%
5906     \glssetabbrvfmt{\glscategory{#1}}%
5907     \Glsaccesssymbol{#1}%
5908   }%
5909 }
```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```
5910 \renewcommand*{\Glossentrysymbol}[1]{%
5911   \glsdoifexistsorwarn{#1}%
5912   {%
5913     \glssetabbrvfmt{\glscategory{#1}}%
5914     \Glsaccesssymbol{#1}%
5915   }%
5916 }
```

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.

```
5917 \newcommand*{\GlsXtrEnableInitialTagging}{%
5918   \@ifstar\s@glsextr@enabletagging\@glsextr@enabletagging
5919 }
5920 \@onlypreamble\GlsXtrEnableInitialTagging
```

`\r@enabletagging` Starred version undefines command.

```
5921 \newcommand*{\s@glsextr@enabletagging}[2]{%
5922   \undef#2%
5923   \@glsextr@enabletagging{#1}{#2}%
5924 }
```

`\r@enabletagging` Internal command.

```
5925 \newcommand*{\@glsextr@enabletagging}[2]{%
    Set attributes for categories given in the first argument.
5926   \@for\@glsextr@cat:=#1\do
5927   {%
```



```

5928 \ifdefempty\@glsxtr@cat
5929 {}%
5930 {\glssetcategoryattribute{\@glsxtr@cat}{tagging}{true}}%
5931 }%
5932 \newrobustcmd*#2[1]{##1}%
5933 \def\@glsxtr@taggingcs{#2}%
5934 \renewcommand*\@glsxtr@activate@initialtagging{%
5935 \let#2\@glsxtr@tag
5936 }%
5937 \ifundef\@gls@preglossaryhook
5938 {\GlossariesExtraWarning{Initial tagging requires at least
5939 glossaries.sty v4.19 to work correctly}}%
5940 {}%
5941 }

```

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

```

5942 \ifundef\mfu@checkword@do
5943 {
5944 \newcommand*{\mfu@checkword@do}[1]{%
5945 \ifdefstring{\mfu@checkword@arg}{#1}%
5946 {%
5947 \let\@mfu@domakefirstuc\@firstofone
5948 \listbreak
5949 }%
5950 }%
5951 }

```

\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.

```

5952 \ifundef\mfu@checkword
5953 {
5954 \newcommand{\@glsxtr@do@titlecaps@warn}{%
5955 \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
5956 support not available}%

```

One warning should suffice.

```

5957 \let\@glsxtr@do@titlecaps@warn\relax
5958 }
5959 }
5960 {
5961 \renewcommand*{\mfu@checkword}[1]{%
5962 \def\mfu@checkword@arg{#1}%
5963 \let\@mfu@domakefirstuc\makefirstuc
5964 \forlistloop\mfu@checkword@do\@mfu@nocaplist
5965 }
5966 }
5967 }

```

```

5968 {}% no patch required

@titlecaps@warn Do warning if title case not supported.
5969 \newcommand*{\@glxtr@do@titlecaps@warn}{}

@initialtagging Used in \printglossary but at least v4.19 of glossaries required.
5970 \newcommand*\@glxtr@activate@initialtagging{}

\@glxtr@tag Definition of tagging command when used in glossary.
5971 \newrobustcmd*{\@glxtr@tag}[1]{%
5972   \glsifattribute{\glscurrententrylabel}{tagging}{true}%
5973   {\glxtrtagfont{#1}}{#1}%
5974 }

\glxtrtagfont Used in the glossary.
5975 \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.
5976 \ifdef\gls@preglossaryhook
5977 {
5978   \renewcommand*\@gls@preglossaryhook{%
5979     \@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.

5980     \ifundef\@glxtr@org@postdescription
5981     {%
5982       \let\@glxtr@org@postdescription\glspostdescription
5983       \renewcommand*\glspostdescription{%
5984         \ifglentryexists{\glscurrententrylabel}%
5985         {%
5986           \glxtrpostdescription
5987           \@glxtr@org@postdescription
5988         }%
5989       }%
5990     }%
5991     }%
5992     {}%

    Enable the options used by \@glxtrp:
5993     \glossxtrsetopts
5994   }%
5995 }
5996 {}

```

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.

```
5997 \newcommand*{\glsxtrpostdescription}{%
5998   \csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%
5999 }
```

postdescgeneral

```
6000 \newcommand*{\glsxtrpostdescgeneral}{}%
```

xtrpostdescterm

```
6001 \newcommand*{\glsxtrpostdescterm}{}%
```

postdescacronym

```
6002 \newcommand*{\glsxtrpostdescacronym}{}%
```

escabbreviation

```
6003 \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.

```
6004 \renewcommand*{\glspostlinkhook}{%
6005   \ifglsentryexists{\glslabel}{\glsxtrpostlinkhook}{}%
6006 }
```

xtrpostlinkhook The entry label should already be stored in `\glslabel` by `\@gls@link`.

```
6007 \newcommand*{\glsxtrpostlinkhook}{%
6008   \glsxtrdiscardperiod{\glslabel}%
6009   {\glsxtrpostlinkendsentence}%
6010   {\glsxtrifcustomdiscardperiod
6011     {\glsxtrifperiod{\glsxtrpostlinkendsentence}{\glsxtrpostlink}}%
6012     {\glsxtrpostlink}%
6013   }%
6014 }
```

omdiscardperiod Allow user to provide a custom check. Should expand to #2 if no check is required otherwise expand to #1.

```
6015 \newcommand*{\glsxtrifcustomdiscardperiod}[2]{#2}
```

\glsxtrpostlink

```
6016 \newcommand*{\glsxtrpostlink}{%
6017   \csuse{glsxtrpostlink\glscategory{\glslabel}}%
6018 }
```

linkendsentence Done by \glxtrpostlinkhook if a full stop is discarded.

```
6019 \newcommand*{\glxtrpostlinkendsentence}{%
6020 \ifcsdef{glxtrpostlink\glscategory{\glslabel}}
6021 {%
6022 \csuse{glxtrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
6023 .\spacefactor\sfcode'\. \relax
6024 }%
6025 {%
```

Assume the full stop was discarded because the entry ends with a period, so adjust the spacefactor.

```
6026 \spacefactor\sfcode'\. \relax
6027 }%
6028 }
```

DescOnFirstUse Provide a command for appending the description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
6029 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
6030 \glxtrifwasfirstuse{\space\glxtrparen{\glssaccessdesc{\glslabel}}}{}%
6031 }
```

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.

```
6032 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
6033 \glxtrifwasfirstuse
6034 {%
6035 \ifglshassymbol{\glslabel}%
6036 {\space\glxtrparen{\glssaccesssymbol{\glslabel}}}%
6037 }%
6038 }%
6039 {}%
6040 }
```

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).

```
6041 \newcommand*{\glxtrdiscardperiod}[3]{%
6042 \glxtrifwasfirstuse
6043 {%
6044 \glusifattribute{#1}{retainfirstuseperiod}{true}%
6045 {#3}%
6046 {%
6047 \glusifattribute{#1}{discardperiod}{true}%
6048 {%
6049 \glusifplural
6050 {%
```

```

6051      \glsifattribute{#1}{pluraldiscardperiod}{true}%
6052      {\glsxtrifperiod{#2}{#3}}%
6053      {#3}%
6054      }%
6055      {%
6056      \glsxtrifperiod{#2}{#3}%
6057      }%
6058      }%
6059      {#3}%
6060      }%
6061  }%
6062  {%
6063  \glsifattribute{#1}{discardperiod}{true}%
6064  {%
6065  \glsifplural
6066  {%
6067  \glsifattribute{#1}{pluraldiscardperiod}{true}%
6068  {\glsxtrifperiod{#2}{#3}}%
6069  {#3}%
6070  }%
6071  {%
6072  \glsxtrifperiod{#2}{#3}%
6073  }%
6074  }%
6075  {#3}%
6076  }%
6077  }

```

`\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`

```
6078 \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 ' ').

```
6079 \newcommand*{\glsxtr@punclist}{.,:;?!}
```

`\punctuationmark` Add character to punctuation list.

```
6080 \newcommand*{\glsxtraddpunctuationmark}[1]{\appto\glsxtr@punclist{#1}}
```

`\punctuationmarks` Reset the punctuation list.

```
6081 \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`.)

```
6082 \newcommand*{\glxtrifnextpunc}[2]{%
6083   \def\reserved@a{#1}%
6084   \def\reserved@b{#2}%
6085   \futurelet\@glspunc@token\glxtr@ifnextpunc
6086 }
```

`glxtr@ifnextpunc`

```
6087 \newcommand*{\glxtr@ifnextpunc}{%
6088   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
6089   \reserved@b
6090 }
```

`glxtr@ifpunctoken` Test if the token given in the first argument is in the punctuation list.

```
6091 \newcommand*{\glxtr@ifpunctoken}[1]{%
6092   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
6093 }
```

`glxtr@ifpunctoken`

```
6094 \def\@glxtr@ifpunctoken#1#2{%
6095   \let\reserved@d=#2%
6096   \ifx\reserved@d\@nnil
6097     \let\glxtr@next\@glxtr@notfoundinlist
6098   \else
6099     \ifx#1\reserved@d
6100       \let\glxtr@next\@glxtr@foundinlist
6101     \else
6102       \let\glxtr@next\@glxtr@ifpunctoken
6103     \fi
6104   \fi
6105   \glxtr@next#1%
6106 }
```

`glxtr@foundinlist`

```
6107 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

`glxtr@notfoundinlist`

```
6108 \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.

```
6109 \newcommand{\glxtr@dopostpunc}[1]{%
6110   \glxtrifnextpunc{\@glxtr@swaptwo{#1}}{#1}%
6111 }
```

@glxtr@swaptwo

```
6112 \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.

```
6113 \define@key{glxtrabbrv}{category}{%
6114   \edef\glscategorylabel{#1}%
6115   \ifcsdef{@glsabbrv@current@#1}%
6116   {%
```

Warning should already have been issued.

```
6117   \let\@glxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
6118   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
6119   \glxtr@applyabbrvstyle{\csname @glsabbrv@current@#1\endcsname}%
6120   \let\GlsXtrWarnDeprecatedAbbrStyle\@glxtr@orgwarndep
6121 }%
6122 {}%
6123 }
```

Save the short plural form. This may be needed before the entry is defined.

```
6124 \define@key{glxtrabbrv}{shortplural}{%
6125   \def\@gl@shortpl{#1}%
6126 }
```

Similarly for the long plural form.

```
6127 \define@key{glxtrabbrv}{longplural}{%
6128   \def\@gl@longpl{#1}%
6129 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```
6130 \newtoks\glsshortpltok
```

\glslongpltok

```
6131 \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.

```

6132 \newcommand*{\@glsxtr@insertdots}[2]{%
6133   \def#1{}%
6134   \@glsxtr@insert@dots#1#2\@nnil
6135 }

```

xtr@insert@dots

```

6136 \newcommand*{\@glsxtr@insert@dots}[2]{%
6137   \ifx\@nnil#2\relax
6138   \let\@glsxtr@insert@dots@next\@gobble
6139   \else
6140   \ifx\relax#2\relax
6141   \else
6142     \appto#1{#2.}%
6143   \fi
6144   \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
6145   \fi
6146   \@glsxtr@insert@dots@next#1%
6147 }

```

Similarly provide a way of replacing spaces with \glsxtrwordsep, which first needs to be defined:

\glsxtrwordsep

```

6148 \newcommand*{\glsxtrwordsep}{\space}

```

Each word is marked with

\glsxtrword

```

6149 \newcommand*{\glsxtrword}[1]{#1}

```

tr@markwordseps

```

6150 \newcommand*{\@glsxtr@markwordseps}[2]{%
6151   \def#1{}%
6152   \@glsxtr@mark@wordseps#1#2 \@nnil
6153 }

```

r@mark@wordseps

```

6154 \def\@glsxtr@mark@wordseps#1#2 #3{%
6155   \ifdefempty{#1}%
6156   {\def#1{\protect\glsxtrword{#2}}}%
6157   {\appto#1{\protect\glsxtrwordsep\protect\glsxtrword{#2}}}%
6158   \ifx\@nnil#3\relax
6159   \let\@glsxtr@mark@wordseps@next\relax
6160   \else
6161   \def\@glsxtr@mark@wordseps@next{%
6162     \@glsxtr@mark@wordseps#1#3}%
6163   \fi
6164   \@glsxtr@mark@wordseps@next
6165 }

```


newabbreviation Define a new generic abbreviation.

```

6166 \newcommand*{\newabbreviation}[4] [] {%
6167   \glsextr@newabbreviation{#1}{#2}{#3}{#4}%
6168 }

```

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.)

```

6169 \newcommand*{\glsextr@newabbreviation}[4] {%
6170   \glscopylisttok{#1}%
6171   \glscopylabeltok{#2}%
6172   \glscopyshorttok{#3}%
6173   \glscopylongtok{#4}%

```

Save the original short and long values (before attribute settings modify them).

```

6174   \def\glsextrorgshort{#3}%
6175   \def\glsextrorglong{#4}%

```

Get the category.

```

6176   \def\glscopycategorylabel{abbreviation}%
6177   \glsextr@applyabbrvstyle{\@glscopyabbrv@current@abbreviation}%

```

Ignore the shortplural and longplural keys.

```

6178   \setkeys*{\glsextrabbrv}{shortplural,longplural}{#1}%

```

Set the default long plural

```

6179   \def\@glscopy@longpl{#4\glscopypluralsuffix}%
6180   \let\@glscopy@default@longpl\@glscopy@longpl

```

Has the markwords attribute been set?

```

6181   \glscopyifcategoryattribute{\glscopycategorylabel}{markwords}{true}%
6182   {%
6183     \@glsextr@markwordseps\@glscopy@long{#4}%
6184     \expandafter\def\expandafter\@glscopy@longpl\expandafter
6185       {\@glscopy@long\glscopypluralsuffix}%
6186     \let\@glscopy@default@longpl\@glscopy@longpl

```

Update \glscopylongtok.

```

6187     \expandafter\glscopylongtok\expandafter{\@glscopy@long}%
6188   }%
6189   {%

```

Has the markshortwords attribute been set? (Not compatible with insertdots.)

```

6190   \glscopyifcategoryattribute{\glscopycategorylabel}{markshortwords}{true}%
6191   {%
6192     \@glsextr@markwordseps\@glscopy@short{#3}%
6193   }%
6194   {%

```

Has the insertdots attribute been set?

```

6195   \glscopyifcategoryattribute{\glscopycategorylabel}{insertdots}{true}%
6196   {%
6197     \@glsextr@insertdots\@glscopy@short{#3}%

```

```

6198      \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
6199    }%
6200    {\def\@gls@short{#3}}%
6201  }%

```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```

6202  \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
6203  {%
6204    \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6205      '\abbrvpluralsuffix}%
6206  }%
6207  {%

```

Has the noshortplural attribute been set?

```

6208  \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
6209  {%
6210    \let\@gls@shortpl\@gls@short
6211  }%
6212  {%
6213    \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6214      \abbrvpluralsuffix}%
6215  }%
6216  }%

```

Update \glsshorttok:

```

6217  \expandafter\glsshorttok\expandafter{\@gls@short}%

```

Hook for further customisation if required:

```

6218  \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).

```

6219  \setkeys*{\glstrabbrv}[category]{#1}%

```

Has the plural been explicitly set?

```

6220  \ifx\@gls@default@longpl\@gls@longpl
6221  \else

```

Has the markwords attribute been set?

```

6222  \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
6223  {%
6224    \expandafter\@glstr@markwordseps\expandafter\@gls@longpl\expandafter
6225    {\@gls@longpl}%
6226  }%
6227  {}%
6228  \fi

```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```

6229  \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
6230  \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

6231  \newabbreviationhook

```

Define this entry:

```

6232 \protected@edef\@do@newglossaryentry{%
6233   \noexpand\newglossaryentry{\the\glslabeltok}%
6234   {%
6235     type=\glstrabbrvtype,%
6236     category=abbreviation,%
6237     short={\the\glsshorttok},%
6238     shortplural={\the\glsshortpltok},%
6239     long={\the\glslongtok},%
6240     longplural={\the\glslongpltok},%
6241     name={\the\glsshorttok},%
6242     \CustomAbbreviationFields,%
6243     \the\glskeylisttok
6244   }%
6245 }%
6246 \@do@newglossaryentry
6247 \GlsXtrPostNewAbbreviation
6248 }

```

evpresetkeyhook Hook for extra stuff in \newabbreviation

```
6249 \newcommand*{\glstrnewabbspresetkeyhook}[3]{}

```

NewAbbreviation Hook used by abbreviation styles.

```
6250 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

bbreviationhook Hook for use with \newabbreviation.

```
6251 \newcommand*{\newabbreviationhook}{}

```

reviationFields

```
6252 \newcommand*{\CustomAbbreviationFields}{}

```

\glstrparen For the parenthetical styles.

```
6253 \newcommand*{\glstrparen}[1]{(#1)}

```

lsxtrfullformat Full format without case change.

```

6254 \newcommand*{\glstrfullformat}[2]{%
6255   \glsfirstlongfont{\glsaccesslong{#1}}#2\glstrfullsep{#1}%
6256   \glstrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
6257 }

```

lsxtrfullformat Full format with case change.

```

6258 \newcommand*{\Glsxtrfullformat}[2]{%
6259   \glsfirstlongfont{\Glsaccesslong{#1}}#2\glstrfullsep{#1}%
6260   \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
6261 }

```

`\glxtrfullplformat` Plural full format without case change.

```
6262 \newcommand*{\glxtrfullplformat}[2]{%
6263   \glsfirstlongfont{\glssaccesslongpl{#1}}#2\glxtrfullsep{#1}%
6264   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{#1}}}%
6265 }
```

`\glxtrfullplformat` Plural full format with case change.

```
6266 \newcommand*{\Glsxtrfullplformat}[2]{%
6267   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glxtrfullsep{#1}%
6268   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6269 }
```

`\glxtrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
6270 \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.

```
6271 \newcommand*{\glxtrinlinefullformat}{\glxtrfullformat}
```

`\Glsxtrinlinefullformat` Full format with case change.

```
6272 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}
```

`\glxtrinlinefullplformat` Plural full format without case change.

```
6273 \newcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}
```

`\Glsxtrinlinefullplformat` Plural full format with case change.

```
6274 \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`

```
6275 \renewcommand*{\gl Sentryfull}[1]{\glxtrinlinefullformat{#1}{}}
```

`\Glsentryfull`

```
6276 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\gl Sentryfullpl`

```
6277 \renewcommand*{\gl Sentryfullpl}[1]{\glxtrinlinefullplformat{#1}{}}
```

`\Glsentryfullpl`

```
6278 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\gl sfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
6279 \newcommand*{\gl sfirstabbrvfont}[1]{\gl sfirstabbrvdefaultfont{#1}}
```

`bbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.
6280 `\newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}`

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.
6281 `\newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}`

`bbrvdefaultfont`
6282 `\newcommand*{\glsabbrvdefaultfont}[1]{#1}`

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.
6283 `\newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}`

`longdefaultfont` Default font changing command used for the long form in commands like `\glsxtrlong`.
6284 `\newcommand*{\glslongdefaultfont}[1]{#1}`

`lsfirstlongfont` Font changing command used for the long form on first use or in the full format.
6285 `\newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}`

`longdefaultfont`
6286 `\newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}`

`brvpluralsuffix` Default plural suffix. Allow an alternative default suffix for abbreviations.
6287 `\newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}`

`brvpluralsuffix` Default plural suffix.
6288 `\newcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}`

`\glsxtrfull` Full form (no case-change).
6289 `\newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\@ns@glsxtrfull}`
6290 `\newcommand*\ns@glsxtrfull[2][\%`
6291 `\new@ifnextchar[{\@glsxtr@full{#1}{#2}}{\%`
6292 `{\@glsxtr@full{#1}{#2}[]}\%`
6293 `}`

`\@glsxtr@full` Low-level macro:
6294 `\def\@glsxtr@full#1#2[#3]{\%`
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).
6295 `\@glsxtr@record{#1}{#2}{\glslink}%`
6296 `\glsdoifexists{#2}%`
6297 `{\%`
6298 `\glssetabbrvfmt{\glscategory{#2}}%`
6299 `\let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper`
6300 `\let\glsifplural\@secondoftwo`
6301 `\let\glsupcase\@firstofthree`
6302 `\let\glsinsert\@empty`
6303 `\def\glscustomtext{\glsxtrinlinefullformat{#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.

```
6304 \glxtrsetupfulldefs
6305 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6306 }%
6307 \glspostlinkhook
6308 }
```

`trsetupfulldefs`

```
6309 \newcommand*\glxtrsetupfulldefs{%
6310 \let\glxtrifwasfirstuse\@firstoftwo
6311 }
```

`\Glsxtrfull` Full form (first letter uppercase).

```
6312 \newrobustcmd*\Glsxtrfull{\@gls@hyp@opt\@ns@Glsxtrfull}
6313 \newcommand*\ns@Glsxtrfull[2][{}]{%
6314 \new@ifnextchar[\@Glsxtr@full{#1}{#2}}%
6315 {\@Glsxtr@full{#1}{#2}[]}%
6316 }
```

`\@Glsxtr@full` Low-level macro:

```
6317 \def\@Glsxtr@full#1#2[#3]{%
6318 \glsdoifexists{#2}%
6319 {%
6320 \glssetabbrvfmt{\glscategory{#2}}%
6321 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6322 \let\glsifplural\@secondoftwo
6323 \let\glsupcase\@secondofthree
6324 \let\glsinsert\@empty
6325 \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
6326 \glxtrsetupfulldefs
6327 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6328 }%
6329 \glspostlinkhook
6330 }
```

`\GLSxtrfull` Full form (all uppercase).

```
6331 \newrobustcmd*\GLSxtrfull{\@gls@hyp@opt\@ns@GLSxtrfull}
6332 \newcommand*\ns@GLSxtrfull[2][{}]{%
6333 \new@ifnextchar[\@GLSxtr@full{#1}{#2}}%
6334 {\@GLSxtr@full{#1}{#2}[]}%
6335 }
```

`\@GLSxtr@full` Low-level macro:

```
6336 \def\@GLSxtr@full#1#2[#3]{%
6337 \glsdoifexists{#2}%
6338 }
```

```

6338 {%
6339   \glsetabbrvfmt{\glscategory{#2}}%
6340   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6341   \let\gl@sifplural\@secondoftwo
6342   \let\glscapscase\@thirdofthree
6343   \let\gl$insert\@empty
6344   \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
6345   \glxtrsetupfulldefs
6346   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6347 }%
6348 \glspostlinkhook
6349 }

```

`\glxtrfullpl` Plural full form (no case-change).

```

6350 \newrobustcmd*{\glxtrfullpl}{\@gl@hyp@opt\@ns@glxtrfullpl}
6351 \newcommand*\ns@glxtrfullpl[2][{}]{%
6352   \new@ifnextchar[{\@glxtr@fullpl{#1}{#2}}%
6353     {\@glxtr@fullpl{#1}{#2}[]}%
6354 }

```

`\@glxtr@fullpl` Low-level macro:

```

6355 \def\@glxtr@fullpl#1#2[#3]{%
  If the record option has been used, the information needs to be written to the aux file regard-
  less of whether the entry exists (unless indexing has been switched off).
6356   \@glxtr@record{#1}{#2}{glslink}%
6357   \glstoifexists{#2}%
6358   {%
6359     \glsetabbrvfmt{\glscategory{#2}}%
6360     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6361     \let\gl@sifplural\@firstoftwo
6362     \let\glscapscase\@firstofthree
6363     \let\gl$insert\@empty
6364     \def\glscustomtext{\glxtrinlinefullplformat{#2}{#3}}%
6365     \glxtrsetupfulldefs
6366     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6367   }%
6368   \glspostlinkhook
6369 }

```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```

6370 \newrobustcmd*{\Glsxtrfullpl}{\@gl@hyp@opt\@ns@Glsxtrfullpl}
6371 \newcommand*\ns@Glsxtrfullpl[2][{}]{%
6372   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6373     {\@Glsxtr@fullpl{#1}{#2}[]}%
6374 }

```

`\@Glsxtr@fullpl` Low-level macro:

```

6375 \def\@Glsxtr@fullpl#1#2[#3]{%

```

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).

```

6376 \@glstr@record{#1}{#2}{glslink}%
6377 \glsoifexists{#2}%
6378 {%
6379 \glsetabbrfmt{\glscategory{#2}}%
6380 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6381 \let\gl@ifplural\@firstoftwo
6382 \let\glscapscase\@secondofthree
6383 \let\glinsert\@empty
6384 \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
6385 \glstrsetupfulldefs
6386 \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
6387 }%
6388 \glspostlinkhook
6389 }

```

\Glsxtrfullpl Plural full form (all upper case).

```

6390 \newrobustcmd*{\Glsxtrfullpl}{\@gl@hyp@opt\ns@Glsxtrfullpl}
6391 \newcommand*{\ns@Glsxtrfullpl}[2][{}]{%
6392 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6393 {\@Glsxtr@fullpl{#1}{#2}[]}%
6394 }

```

\@Glsxtr@fullpl Low-level macro:

```

6395 \def\@Glsxtr@fullpl#1#2[#3]{%

```

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).

```

6396 \@glstr@record{#1}{#2}{glslink}%
6397 \glsoifexists{#2}%
6398 {%
6399 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6400 \let\gl@ifplural\@firstoftwo
6401 \let\glscapscase\@thirdofthree
6402 \let\glinsert\@empty
6403 \def\glscustomtext{%
6404 \mfirstucMakeUppercase{\glxtrinlinefullplformat{#2}{#3}}}%
6405 \glstrsetupfulldefs
6406 \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
6407 }%
6408 \glspostlinkhook
6409 }

```

The short and long forms work in a similar way to acronyms.

\glstrshort

```

6410 \newrobustcmd*{\glstrshort}{\@gl@hyp@opt\ns@glstrshort}

```


Define the un-starred form. Need to determine if there is a final optional argument

```
6411 \newcommand*{\ns@glstrshort}[2] [] {%
6412   \new@ifnextchar[{\@glstrshort{#1}{#2}}{\@glstrshort{#1}{#2} []}%
6413 }
```

Read in the final optional argument:

```
6414 \def\@glstrshort#1#2[#3] {%
```

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).

```
6415   \@glstr@record{#1}{#2}{glslink}%
6416   \glstoifexists{#2}%
6417   {%
```

Need to make sure \glssabrvfont is set correctly.

```
6418     \glsssetabrvfmt{\glscategory{#2}}%
6419     \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
6420     \let\glstrifwasfirstuse\@secondoftwo
6421     \let\glslifplural\@secondoftwo
6422     \let\glscapscase\@firstofthree
6423     \let\glslinsert\@empty
6424     \def\glscustomtext{%
6425       \glssabrvfont{\glssaccessshort{#2}\ifglstrinsertinside#3\fi}%
6426       \ifglstrinsertinside\else#3\fi
6427     }%
6428     \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6429   }%
6430   \glspostlinkhook
6431 }
```

\Glsxtrshort

```
6432 \newrobustcmd*{\Glsxtrshort}{\@glshyp@opt\ns@Glsxtrshort}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6433 \newcommand*{\ns@Glsxtrshort}[2] [] {%
6434   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} []}%
6435 }
```

Read in the final optional argument:

```
6436 \def\@Glsxtrshort#1#2[#3] {%
```

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).

```
6437   \@glstr@record{#1}{#2}{glslink}%
6438   \glstoifexists{#2}%
6439   {%
6440     \glsssetabrvfmt{\glscategory{#2}}%
6441     \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
6442     \let\glstrifwasfirstuse\@secondoftwo
6443     \let\glslifplural\@secondoftwo
6444     \let\glscapscase\@secondofthree
```

```

6445 \let\glsinsert\@empty
6446 \def\glscustomtext{%
6447   \glsabbrvfont{\Glsaccessshort{#2}\ifglxtrinsertinside#3\fi}%
6448   \ifglxtrinsertinside\else#3\fi
6449 }%
6450 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6451 }%
6452 \glspostlinkhook
6453 }

```

\GLSxtrshort

```

6454 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6455 \newcommand*{\ns@GLSxtrshort}[2][\%
6456   \new@ifnextchar[\@GLSxtrshort{#1}{#2}]{\@GLSxtrshort{#1}{#2}[\%
6457 ]}

```

Read in the final optional argument:

```

6458 \def\@GLSxtrshort#1#2[#3]{%

```

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).

```

6459 \@glsxtr@record{#1}{#2}{\glslink}%
6460 \glsdoifexists{#2}%
6461 {%
6462   \glssetabbrvfmt{\glscategory{#2}}%
6463   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6464   \let\glsxtrifwasfirstuse\@secondoftwo
6465   \let\glsifplural\@secondoftwo
6466   \let\glsapscase\@thirdofthree
6467   \let\glsinsert\@empty
6468   \def\glscustomtext{%
6469     \mfirstucMakeUppercase
6470     {\glsabbrvfont{\Glsaccessshort{#2}\ifglxtrinsertinside#3\fi}%
6471     \ifglxtrinsertinside\else#3\fi
6472   }%
6473   }%
6474   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6475 }%
6476 \glspostlinkhook
6477 }

```

\glsxtrlong

```

6478 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6479 \newcommand*{\ns@glsxtrlong}[2][\%
6480   \new@ifnextchar[\@glsxtrlong{#1}{#2}]{\@glsxtrlong{#1}{#2}[\%
6481 ]}

```

Read in the final optional argument:

```
6482 \def\@glxstrlong#1#2[#3]{%
```

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).

```
6483 \@glxstr@record{#1}{#2}{glslink}%
6484 \glstoifexists{#2}%
6485 {%
6486 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6487 \let\glxstrifwasfirstuse\@secondoftwo
6488 \let\gl@ifplural\@secondoftwo
6489 \let\glscapscase\@firstofthree
6490 \let\glinsert\@empty
6491 \def\glscustomtext{%
6492 \glslongfont{\glaccesslong{#2}\ifglxtrininsertinside#3\fi}%
6493 \ifglxtrininsertinside\else#3\fi
6494 }%
6495 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6496 }%
6497 \glspostlinkhook
6498 }
```

\Glsxtrlong

```
6499 \newrobustcmd*{\Glsxtrlong}{\@gl@hyp@opt\@ns@Glsxtrlong}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6500 \newcommand*{\ns@Glsxtrlong}[2][{}]{%
6501 \new@ifnextchar[\@Glsxtrlong{#1}{#2}]{\@Glsxtrlong{#1}{#2}[{}]}%
6502 }
```

Read in the final optional argument:

```
6503 \def\@Glsxtrlong#1#2[#3]{%
```

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).

```
6504 \@glxstr@record{#1}{#2}{glslink}%
6505 \glstoifexists{#2}%
6506 {%
6507 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6508 \let\glxstrifwasfirstuse\@secondoftwo
6509 \let\gl@ifplural\@secondoftwo
6510 \let\glscapscase\@secondofthree
6511 \let\glinsert\@empty
6512 \def\glscustomtext{%
6513 \glslongfont{\Glsaccesslong{#2}\ifglxtrininsertinside#3\fi}%
6514 \ifglxtrininsertinside\else#3\fi
6515 }%
6516 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6517 }%
6518 \glspostlinkhook
6519 }
```

\GLSxtrlong

```
6520 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\@ns@GLSxtrlong}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6521 \newcommand*{\ns@GLSxtrlong}[2] [] {%
```

```
6522   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
```

```
6523 }
```

Read in the final optional argument:

```
6524 \def\@GLSxtrlong#1#2[#3] {%
```

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).

```
6525   \@glsxtr@record{#1}{#2}{glslink}%
```

```
6526   \glsdoifexists{#2}%
```

```
6527   {%
```

```
6528     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
6529     \let\glsxtrifwasfirstuse\@secondoftwo
```

```
6530     \let\glsifplural\@secondoftwo
```

```
6531     \let\glscapscase\@thirdofthree
```

```
6532     \let\glsinsert\@empty
```

```
6533     \def\glscustomtext{%
```

```
6534       \mfirstucMakeUppercase
```

```
6535       {\glsfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
```

```
6536       \ifglsxtrininsertinside\else#3\fi
```

```
6537     }%
```

```
6538   }%
```

```
6539   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
```

```
6540 }%
```

```
6541 \glspostlinkhook
```

```
6542 }
```

Plural short forms:

\glsxtrshortpl

```
6543 \newrobustcmd*{\glsxtrshortpl}{\@gls@hyp@opt\@ns@glsxtrshortpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6544 \newcommand*{\ns@glsxtrshortpl}[2] [] {%
```

```
6545   \new@ifnextchar[{\@glsxtrshortpl{#1}{#2}}{\@glsxtrshortpl{#1}{#2} []}%
```

```
6546 }
```

Read in the final optional argument:

```
6547 \def\@glsxtrshortpl#1#2[#3] {%
```

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).

```
6548   \@glsxtr@record{#1}{#2}{glslink}%
```

```
6549   \glsdoifexists{#2}%
```

```
6550   {%
```

```
6551     \glsssetabbrvfmt{\glscategory{#2}}%
```

```

6552 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6553 \let\glxtrifwasfirstuse\@secondoftwo
6554 \let\gl@ifplural\@firstoftwo
6555 \let\glscapscase\@firstofthree
6556 \let\glinsert\@empty
6557 \def\glscustomtext{%
6558     \glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
6559     \ifglxtrininsertinside\else#3\fi
6560 }%
6561 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6562 }%
6563 \glspostlinkhook
6564 }

```

\Glsxtrshortpl

```

6565 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\ns@Glsxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6566 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
6567     \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
6568 }

```

Read in the final optional argument:

```

6569 \def\@Glsxtrshortpl#1#2[#3] {%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
6570 \@glxtr@record{#1}{#2}{\gl@link}%
6571 \gl@ifexists{#2}%
6572 {%
6573     \glsetabbrvfmt{\glcategory{#2}}%
6574     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6575     \let\glxtrifwasfirstuse\@secondoftwo
6576     \let\gl@ifplural\@firstoftwo
6577     \let\glscapscase\@secondofthree
6578     \let\glinsert\@empty
6579     \def\glscustomtext{%
6580         \glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
6581         \ifglxtrininsertinside\else#3\fi
6582     }%
6583     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6584 }%
6585 \glspostlinkhook
6586 }

```

\GLSxtrshortpl

```

6587 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6588 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%

```

```

6589 \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
6590 }

```

Read in the final optional argument:

```

6591 \def\@GLSxtrshortpl#1#2[#3]{%

```

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).

```

6592 \@glxtr@record{#1}{#2}{glslink}%
6593 \glsoifexists{#2}%
6594 {%
6595   \glsetabbrvfmt{\glscategory{#2}}%
6596   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6597   \let\glxtrifwasfirstuse\@secondoftwo
6598   \let\gl@ifplural\@firstoftwo
6599   \let\glscapscase\@thirdofthree
6600   \let\glinsert\@empty
6601   \def\glscustomtext{%
6602     \mfirstucMakeUppercase
6603     {\glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
6604     \ifglxtrininsertinside\else#3\fi
6605   }%
6606   }%
6607   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6608   }%
6609   \glspostlinkhook
6610 }

```

Plural long forms:

\glxtrlongpl

```

6611 \newrobustcmd*{\glxtrlongpl}{\@gl@hyp@opt\@ns@glxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6612 \newcommand*{\ns@glxtrlongpl}[2] [] {%
6613   \new@ifnextchar[{\@glxtrlongpl{#1}{#2}}{\@glxtrlongpl{#1}{#2} []}%
6614 }

```

Read in the final optional argument:

```

6615 \def\@glxtrlongpl#1#2[#3]{%

```

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).

```

6616 \@glxtr@record{#1}{#2}{glslink}%
6617 \glsoifexists{#2}%
6618 {%
6619   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6620   \let\glxtrifwasfirstuse\@secondoftwo
6621   \let\gl@ifplural\@firstoftwo
6622   \let\glscapscase\@firstofthree
6623   \let\glinsert\@empty

```

```

6624 \def\glscustomtext{%
6625 \glslongfont{\glsaccesslongpl{#2}\ifglxtrinsertinside#3\fi}%
6626 \ifglxtrinsertinside\else#3\fi
6627 }%
6628 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6629 }%
6630 \glspostlinkhook
6631 }

```

\Glsxtrlongpl

```

6632 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6633 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
6634 \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}]%
6635 }

```

Read in the final optional argument:

```

6636 \def\@Glsxtrlongpl#1#2[#3]{%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
6637 \@glsxtr@record{#1}{#2}{\glslink}%
6638 \glsdoifexists{#2}%
6639 {%
6640 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6641 \let\glsxtrifwasfirstuse\@secondoftwo
6642 \let\glsifplural\@firstoftwo
6643 \let\glscapscase\@secondofthree
6644 \let\glsinsert\@empty
6645 \def\glscustomtext{%
6646 \glslongfont{\Glsaccesslongpl{#2}\ifglxtrinsertinside#3\fi}%
6647 \ifglxtrinsertinside\else#3\fi
6648 }%
6649 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6650 }%
6651 \glspostlinkhook
6652 }

```

\GLSxtrlongpl

```

6653 \newrobustcmd*{\GLSxtrlongpl}{\@gls@hyp@opt\@ns@GLSxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6654 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
6655 \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}]%
6656 }
    Read in the final optional argument:
6657 \def\@GLSxtrlongpl#1#2[#3]{%

```

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).

```

6658 \@glstr@record{#1}{#2}{glslink}%
6659 \glsoifexists{#2}%
6660 {%
6661 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6662 \let\glstrifwasfirstuse\@secondoftwo
6663 \let\gl@ifplural\@firstoftwo
6664 \let\glscapscase\@thirdofthree
6665 \let\glinsert\@empty
6666 \def\glscustomtext{%
6667 \mfirstucMakeUppercase
6668 {\glslongfont{\glaccesslongpl{#2}\ifglstrinsertinside#3\fi}%
6669 \ifglstrinsertinside\else#3\fi
6670 }%
6671 }%
6672 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6673 }%
6674 \glspostlinkhook
6675 }

```

\glsetabbrvfmt Set the current format for the given category (or the abbreviation category if unset).

```

6676 \newcommand*{\glsetabbrvfmt}[1]{%
6677 \ifcsdef{@glsabbrv@current@#1}%
6678 {\glstr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
6679 {\glstr@applyabbrvfmt{@glsabbrv@current@abbreviation}}%
6680 }

```

\gluseabbrvfont Provide a way to use the abbreviation font for a given category for arbitrary text.

```

6681 \newrobustcmd*{\gluseabbrvfont}[2]{\glsetabbrvfmt{#2}\glsabbrvfont{#1}}

```

\gluselongfont Provide a way to use the long font for a given category for arbitrary text.

```

6682 \newrobustcmd*{\gluselongfont}[2]{\glsetabbrvfmt{#2}\glslongfont{#1}}

```

\glstrgenabbrvfmt Similar to `\glsgenacfmt`, but for abbreviations.

```

6683 \newcommand*{\glstrgenabbrvfmt}{%
6684 \ifdefempty\glscustomtext
6685 {%
6686 \ifglused\glslabel
6687 {%

```

Subsequent use:

```

6688 \gl@ifplural
6689 {%

```

Subsequent plural form:

```

6690 \glscapscase
6691 {%

```


Subsequent plural form, don't adjust case:

```
6692      \glxtrsubsequentplfmt{\glslabel}{\glsinsert}%  
6693      }%  
6694      {%
```

Subsequent plural form, make first letter upper case:

```
6695      \Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%  
6696      }%  
6697      {%
```

Subsequent plural form, all caps:

```
6698      \mfirstucMakeUppercase  
6699      {\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%  
6700      }%  
6701      }%  
6702      {%
```

Subsequent singular form

```
6703      \glscapscase  
6704      {%
```

Subsequent singular form, don't adjust case:

```
6705      \glxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6706      }%  
6707      {%
```

Subsequent singular form, make first letter upper case:

```
6708      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6709      }%  
6710      {%
```

Subsequent singular form, all caps:

```
6711      \mfirstucMakeUppercase  
6712      {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%  
6713      }%  
6714      }%  
6715      }%  
6716      {%
```

First use:

```
6717      \glsifplural  
6718      {%
```

First use plural form:

```
6719      \glscapscase  
6720      {%
```

First use plural form, don't adjust case:

```
6721      \glxtrfullplformat{\glslabel}{\glsinsert}%  
6722      }%  
6723      {%
```

First use plural form, make first letter upper case:

```
6724      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
6725      }%
6726      {%
```

First use plural form, all caps:

```
6727      \mfirstucMakeUppercase
6728      {\glsxtrfullplformat{\glslabel}{\glsinsert}}%
6729      }%
6730      }%
6731      {%
```

First use singular form

```
6732      \glscapscase
6733      {%
```

First use singular form, don't adjust case:

```
6734      \glsxtrfullformat{\glslabel}{\glsinsert}%
6735      }%
6736      {%
```

First use singular form, make first letter upper case:

```
6737      \Glsxtrfullformat{\glslabel}{\glsinsert}%
6738      }%
6739      {%
```

First use singular form, all caps:

```
6740      \mfirstucMakeUppercase
6741      {\glsxtrfullformat{\glslabel}{\glsinsert}}%
6742      }%
6743      }%
6744      }%
6745      }%
6746      {%
```

User supplied text.

```
6747      \glscustomtext
6748      }%
6749 }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
6750 \newcommand*{\glsxtrsubsequentfmt}[2]{%
6751   \glsabbrvfont{\glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%
6752   \ifglsxtrininsertinside \else#2\fi
6753 }
6754 \let\glsxtrdefaultsubsequentfmt\glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural no case change).

```
6755 \newcommand*{\glsxtrsubsequentplfmt}[2]{%
6756   \glsabbrvfont{\glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%
6757   \ifglsxtrininsertinside \else#2\fi
```

```

6758 }
6759 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt

trsubsequentfmt Subsequent use format (singular, first letter uppercase).
6760 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
6761   \glsabbrvfont{\Glsaccesssshort{#1}\ifglxtrininsertinside #2\fi}%
6762   \ifglxtrininsertinside \else#2\fi
6763 }
6764 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt

subsequentplfmt Subsequent use format (plural, first letter uppercase).
6765 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
6766   \glsabbrvfont{\Glsaccesssshortpl{#1}\ifglxtrininsertinside #2\fi}%
6767   \ifglxtrininsertinside \else#2\fi
6768 }
6769 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt

```

1.7.1 Abbreviation Styles Setup

```

breviationstyle
6770 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
6771   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
6772   {%
6773     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
6774   }%
6775   {%

Have abbreviations already been defined for this category?
6776   \ifcsstring{@glsabbrv@current@#1}{#2}%
6777   {%

Style already set.
6778   }%
6779   {%
6780     \def\@glxtr@dostylewarn{%
6781       \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
6782       {%
6783         \def\@glxtr@dostylewarn{\GlossariesWarning{Abbreviation
6784           style has been switched \MessageBreak
6785           for category ‘#1’, \MessageBreak
6786           but there have already been entries \MessageBreak
6787           defined for this category. Unwanted \MessageBreak
6788           side-effects may result}}%
6789         \@endfortrue
6790       }%
6791       \@glxtr@dostylewarn

Set up the style for the given category.
6792       \csdef{@glsabbrv@current@#1}{#2}%
6793       \glxtr@applyabbrvstyle{#2}%

```

```

6794     }%
6795 }%
6796 }

```

`\applyabbrvstyle` Apply the abbreviation style without existence check.

```

6797 \newcommand*{\glxtr@applyabbrvstyle}[1]{%
6798   \csuse{@glsabbrv@dispstyle@setup@#1}%
6799   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6800 }

```

`\r@applyabbrvfmt` Only apply the style formats.

```

6801 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
6802   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6803 }

```

`\abbreviationstyle` This is different from `\newacronymstyle`. The first argument is the label, the second argument sets the information required when defining the new abbreviation and the third argument sets the commands used to display the full format.

```

6804 \newcommand*{\newabbreviationstyle}[3]{%
6805   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
6806   {%
6807     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
6808       defined}{}%
6809   }%
6810   {%
6811     \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```

6812     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6813     #2}%
6814     \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```

6815     \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
6816     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6817     \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
6818     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

Reset `\glxtrsubsequentfmt` etc in case a style changes this.

```

6819     \let\glxtrsubsequentfmt\glxtrdefaultsubsequentfmt
6820     \let\glxtrsubsequentplfmt\glxtrdefaultsubsequentplfmt
6821     \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
6822     \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
6823     #3}%
6824   }%
6825 }

```

`\abbreviationstyle`

```

6826 \newcommand*{\renewabbreviationstyle}[3]{%
6827   \ifcsundef{@glsabbrv@dispstyle@setup@#1}

```

```

6828 {%
6829   \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
6830 }%
6831 {%
6832   \csdef{@glsabbrv@dispstyle@setup@#1}{%
      Initialise hook to do nothing. The style may change this.
6833     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6834     #2}%
6835   \csdef{@glsabbrv@dispstyle@fmts@#1}{%
      Assume in-line form is the same as first use. The style may change this.
6836     \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6837     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6838     \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6839     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
6840     #3}%
6841   }%
6842 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

6843 \newcommand*{\letabbreviationstyle}[2]{%
6844   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
6845   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6846 }

```

deprecated@abbrstyle `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

6847 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
6848   \csdef{@glsabbrv@dispstyle@setup@#1}{%
6849     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
6850     \csuse{@glsabbrv@dispstyle@setup@#2}%
6851   }%
6852   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6853 }

```

deprecatedAbbrStyle Generate warning for deprecated style use.

```

6854 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
6855   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
6856     use ‘#2’ instead}%
6857 }

```

eAbbrStyleSetup

```

6858 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
6859   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%

```

```

6860 {%
6861     \PackageError{glossaries-extra}%
6862     {Unknown abbreviation style definitions ‘#1’}{}%
6863 }%
6864 {%
6865     \csname @glsabbrv@dispstyle@setup@#1\endcsname
6866 }%
6867 }

```

seAbbrStyleFmts

```

6868 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
6869     \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
6870     {%
6871         \PackageError{glossaries-extra}%
6872         {Unknown abbreviation style formats ‘#1’}{}%
6873     }%
6874     {%
6875         \csname @glsabbrv@dispstyle@fmts@#1\endcsname
6876     }%
6877 }

```

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.

```

6878 \newif\ifglsxtrinsertinside
6879 \glsxtrinsertinsidefalse

```

trlongshortname

```

6880 \newcommand*{\glsxtrlongshortname}{%
6881     \protect\glsabbrvfont{\the\glsshorttok}%
6882 }

```

long-short

```

6883 \newabbreviationstyle{long-short}%
6884 {%
6885     \renewcommand*{\CustomAbbreviationFields}{%
6886         name={\glsxtrlongshortname},
6887         sort={\the\glsshorttok},

```

```

6888 first={\protect\glsfirstlongfont{\the\glslongtok}%
6889 \protect\glsextrfullsep{\the\glslabeltok}%
6890 \glsextrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6891 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6892 \protect\glsextrfullsep{\the\glslabeltok}%
6893 \glsextrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%

6894 plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6895 description={\the\glslongtok}}%

```

Unset the regular attribute if it has been set.

```

6896 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6897 \glsasattribute{\the\glslabeltok}{regular}%
6898 {%
6899 \glssetattribute{\the\glslabeltok}{regular}{false}%
6900 }%
6901 {}%
6902 }%
6903 }%
6904 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6905 \renewcommand*{\abbrvpluralsuffix}{\glsextrabbrvpluralsuffix}%
6906 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6907 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6908 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6909 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6910 \renewcommand*{\glsextrfullformat}[2]{%
6911 \glsfirstlongfont{\glsaccesslong{##1}\ifglsextrininsertinside##2\fi}%
6912 \ifglsextrininsertinside\else##2\fi
6913 \glsextrfullsep{##1}%
6914 \glsextrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6915 }%
6916 \renewcommand*{\glsextrfullplformat}[2]{%
6917 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsextrininsertinside##2\fi}%
6918 \ifglsextrininsertinside\else##2\fi\glsextrfullsep{##1}%
6919 \glsextrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6920 }%
6921 \renewcommand*{\Glsxtrfullformat}[2]{%
6922 \glsfirstlongfont{\Glsaccesslong{##1}\ifglsextrininsertinside##2\fi}%
6923 \ifglsextrininsertinside\else##2\fi\glsextrfullsep{##1}%
6924 \glsextrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6925 }%
6926 \renewcommand*{\Glsxtrfullplformat}[2]{%
6927 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsextrininsertinside##2\fi}%
6928 \ifglsextrininsertinside\else##2\fi\glsextrfullsep{##1}%
6929 \glsextrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6930 }%
6931 }

```

Set this as the default style for general abbreviations:

```
6932 \setabbreviationstyle{long-short}
```

ngshortdescsort

```
6933 \newcommand*{\glxtrlongshortdescsort}{%
6934 \expandonce\glxtrorglong\space (\expandonce\glxtrorgshort)%
6935 }
```

ngshortdescname

```
6936 \newcommand*{\glxtrlongshortdescname}{%
6937 \protect\glslongfont{\the\glslongtok}
6938 \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}}%
6939 }
```

long-short-desc User supplies description. The long form is included in the name.

```
6940 \newabbreviationstyle{long-short-desc}%
6941 {%
6942 \renewcommand*{\CustomAbbreviationFields}{%
6943 name={\glxtrlongshortdescname},
6944 sort={\glxtrlongshortdescsort},%
6945 first={\protect\glsfirstlongfont{\the\glslongtok}%
6946 \protect\glxtrfullsep{\the\glslabeltok}%
6947 \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6948 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6949 \protect\glxtrfullsep{\the\glslabeltok}%
6950 \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
```

The text key should only have the short form.

```
6951 text={\protect\glsabbrvfont{\the\glsshorttok}},%
6952 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
6953 }%
```

Unset the regular attribute if it has been set.

```
6954 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6955 \glshasattribute{\the\glslabeltok}{regular}%
6956 {%
6957 \glssetattribute{\the\glslabeltok}{regular}{false}%
6958 }%
6959 {}%
6960 }%
6961 }%
6962 {%
6963 \GlsXtrUseAbbrStyleFmts{long-short}%
6964 }
```

trshortlongname

```
6965 \newcommand*{\glxtrshortlongname}{%
6966 \protect\glsabbrvfont{\the\glsshorttok}%
6967 }
```


short-long Short form followed by long form in parenthesis on first use.

```
6968 \newabbreviationstyle{short-long}%
6969 {%
6970   \renewcommand*{\CustomAbbreviationFields}{%
6971     name={\glxtrshortlongname},
6972     sort={\the\glsshorttok},
6973     description={\the\gslongtok},%
6974     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6975       \protect\glxtrfullsep{\the\glslabeltok}%
6976       \glxtrparen{\protect\glsfirstlongfont{\the\gslongtok}}},%
6977     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6978       \protect\glxtrfullsep{\the\glslabeltok}%
6979       \glxtrparen{\protect\glsfirstlongfont{\the\gslongpltok}}},%
6980     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6981 }
```

Unset the regular attribute if it has been set.

```
6981 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6982   \glshasattribute{\the\glslabeltok}{regular}%
6983   {%
6984     \glissetattribute{\the\glslabeltok}{regular}{false}%
6985   }%
6986   {}%
6987 }%
6988 }%
6989 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6990 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
6991 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6992 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6993 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6994 \renewcommand*{\gslongfont}[1]{\gslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
6995 \renewcommand*{\glxtrfullformat}[2]{%
6996   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6997   \ifglxtrininsertinside\else##2\fi
6998   \glxtrfullsep{##1}%
6999   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7000 }%
7001 \renewcommand*{\glxtrfullplformat}[2]{%
7002   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7003   \ifglxtrininsertinside\else##2\fi
7004   \glxtrfullsep{##1}%
7005   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7006 }%
7007 \renewcommand*{\Glsxtrfullformat}[2]{%
7008   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7009   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7010 }
```

```

7010 \glstrparen{\glfirstlongfont{\glaccesslong{##1}}}%
7011 }%
7012 \renewcommand*{\Glsxtrfullplformat}[2]{%
7013 \glfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7014 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7015 \glstrparen{\glfirstlongfont{\glaccesslongpl{##1}}}%
7016 }%
7017 }

```

ortlongdescsort

```

7018 \newcommand*{\glxtrshortlongdescsort}{\the\glsshorttok}

```

ortlongdescname

```

7019 \newcommand*{\glxtrshortlongdescname}{%
7020 \protect\glsabbrvfont{\the\glsshorttok}
7021 \glstrparen{\protect\glslongfont{\the\glslongtok}}}%
7022 }

```

short-long-desc User supplies description. The long form is included in the name.

```

7023 \newabbreviationstyle{short-long-desc}%
7024 {%
7025 \renewcommand*{\CustomAbbreviationFields}{%
7026 name={\glxtrshortlongdescname},
7027 sort={\glxtrshortlongdescsort},
7028 first={\protect\glfirstabbrvfont{\the\glsshorttok}}%
7029 \protect\glxtrfullsep{\the\glslabeltok}}%
7030 \glstrparen{\protect\glfirstlongfont{\the\glslongtok}}},%
7031 firstplural={\protect\glfirstabbrvfont{\the\glsshortpltok}}%
7032 \protect\glxtrfullsep{\the\glslabeltok}}%
7033 \glstrparen{\protect\glfirstlongfont{\the\glslongpltok}}},%
7034 text={\protect\glsabbrvfont{\the\glsshorttok}},%
7035 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
7036 }%

```

Unset the regular attribute if it has been set.

```

7037 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7038 \glshasattribute{\the\glslabeltok}{regular}%
7039 {%
7040 \glissetattribute{\the\glslabeltok}{regular}{false}%
7041 }%
7042 {}%
7043 }%
7044 }%
7045 {%
7046 \GlsXtrUseAbbrStyleFmts{short-long}%
7047 }

```

ongfootnotefont Only used by the “footnote” styles.

```
7048 \newcommand*{\glfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
7049 \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`).

```
7050 \newcommand*{\glxtrabbrvfootnote}[2]{\footnote{#2}}
```

xtrfootnotename

```
7051 \newcommand*{\glxtrfootnotename}{%  
7052   \protect\glsabbrvfont{\the\glsshorttok}%  
7053 }
```

footnote Short form followed by long form in footnote on first use.

```
7054 \newabbreviationstyle{footnote}%  
7055 {%  
7056   \renewcommand*{\CustomAbbreviationFields}{%  
7057     name={\glxtrfootnotename},  
7058     sort={\the\glsshorttok},  
7059     description={\the\glslongtok},%  
  
7060     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%  
7061     \protect\glxtrabbrvfootnote{\the\glslabeltok}%  
7062     {\protect\glfirstlongfootnotefont{\the\glslongtok}}},%  
7063     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%  
7064     \protect\glxtrabbrvfootnote{\the\glslabeltok}%  
7065     {\protect\glfirstlongfootnotefont{\the\glslongpltok}}},%  
  
7066     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.

```
7067   \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
7068     \glsssetAttribute{\the\glslabeltok}{nohyperfirst}{true}%  
7069     \glshasattribute{\the\glslabeltok}{regular}%  
7070     {%  
7071       \glsssetAttribute{\the\glslabeltok}{regular}{false}%  
7072     }%  
7073   {}%  
7074 }
```

7075 }%
 7076 {%

In case the user wants to mix and match font styles, these are redefined here.

```
7077 \renewcommand*\abbrvpluralsuffix{\glstrabbrvpluralsuffix}%
7078 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
7079 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
7080 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
7081 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
7082 \renewcommand*\glstrfullformat[2]{%
7083   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
7084   \ifglstrinsertinside\else##2\fi
7085   \protect\glstrabbrvfootnote{##1}%
7086   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7087 }%
7088 \renewcommand*\glstrfullplformat[2]{%
7089   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
7090   \ifglstrinsertinside\else##2\fi
7091   \protect\glstrabbrvfootnote{##1}%
7092   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7093 }%
7094 \renewcommand*\Glsxtrfullformat[2]{%
7095   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
7096   \ifglstrinsertinside\else##2\fi
7097   \protect\glstrabbrvfootnote{##1}%
7098   {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7099 }%
7100 \renewcommand*\Glsxtrfullplformat[2]{%
7101   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
7102   \ifglstrinsertinside\else##2\fi
7103   \protect\glstrabbrvfootnote{##1}%
7104   {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
7105 }%
```

The first use full form and the inline full form use the short (long) style.

```
7106 \renewcommand*\glstrinlinefullformat[2]{%
7107   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
7108   \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
7109   \glstrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7110 }%
7111 \renewcommand*\glstrinlinefullplformat[2]{%
7112   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
7113   \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
7114   \glstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7115 }%
7116 \renewcommand*\Glsxtrinlinefullformat[2]{%
7117   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
7118   \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
7119   \glstrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7120 }
```

```

7120 }%
7121 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7122   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7123   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7124   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
7125 }%
7126 }

```

short-footnote

```

7127 \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.

```

7128 \newabbreviationstyle{postfootnote}%
7129 {%
7130   \renewcommand*{\CustomAbbreviationFields}{%
7131     name={\glxtrfootnotename},
7132     sort={\the\glsshorttok},
7133     description={\the\glslongtok},%
7134     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
7135     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
7136     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

7137 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7138   \csdef{glxtrpostlink\glscategorylabel}{%
7139     \glxtrifwasfirstuse
7140     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

7141       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
7142       {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
7143     }%
7144   }%
7145 }%
7146 \glshasattribute{\the\glslabeltok}{regular}%
7147 {%
7148   \glissetattribute{\the\glslabeltok}{regular}{false}%
7149 }%
7150 {}%
7151 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

7152 \renewcommand*{\glxtrsetupfulldefs}{%
7153   \let\glxtrifwasfirstuse\@secondoftwo

```

```

7154 }%
7155 }%
7156 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7157 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
7158 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
7159 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
7160 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
7161 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7162 \renewcommand*\glxtrfullformat[2]{%
7163   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7164   \ifglxtrininsertinside\else##2\fi
7165 }%
7166 \renewcommand*\glxtrfullplformat[2]{%
7167   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7168   \ifglxtrininsertinside\else##2\fi
7169 }%
7170 \renewcommand*\Glsxtrfullformat[2]{%
7171   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7172   \ifglxtrininsertinside\else##2\fi
7173 }%
7174 \renewcommand*\Glsxtrfullplformat[2]{%
7175   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7176   \ifglxtrininsertinside\else##2\fi
7177 }%

```

The first use full form and the inline full form use the short (long) style.

```

7178 \renewcommand*\glxtrininlinefullformat[2]{%
7179   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7180   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7181   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7182 }%
7183 \renewcommand*\glxtrininlinefullplformat[2]{%
7184   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7185   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7186   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7187 }%
7188 \renewcommand*\Glsxtrininlinefullformat[2]{%
7189   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7190   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7191   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7192 }%
7193 \renewcommand*\Glsxtrininlinefullplformat[2]{%
7194   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7195   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7196   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
7197 }%
7198 }

```

rt-postfootnote

```
7199 \letabbreviationstyle{short-postfootnote}{postfootnote}
```

shortnolongname

```
7200 \newcommand*{\glxtrshortnolongname}{%
7201   \protect\glsabbrvfont{\the\glsshorttok}%
7202 }
```

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.

```
7203 \newabbreviationstyle{short}%
7204 {%
7205   \renewcommand*{\CustomAbbreviationFields}{%
7206     name={\glxtrshortnolongname},
7207     sort={\the\glsshorttok},
7208     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
7209     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
7210     text={\protect\glsabbrvfont{\the\glsshorttok}},
7211     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7212     description={\the\glslongtok}}%
7213   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7214     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7215 }%
7216 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
7217 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7218 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7219 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7220 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7221 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
7222 \renewcommand*{\glxtrinlinefullformat}[2]{%
7223   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
7224   \ifglxtrininsertinside##2\fi}%
7225   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7226   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7227 }%
7228 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7229   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
7230   \ifglxtrininsertinside##2\fi}%
7231   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7232   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7233 }%
7234 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7235   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
```

```

7236     \ifglxtrinsertinside##2\fi}%
7237     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7238     \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
7239 }%
7240 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7241     \protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}%
7242         \ifglxtrinsertinside##2\fi}%
7243     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7244     \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7245 }%

```

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.

```

7246 \renewcommand*{\glxtrfullformat}[2]{%
7247     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7248     \ifglxtrinsertinside\else##2\fi
7249 }%
7250 \renewcommand*{\glxtrfullplformat}[2]{%
7251     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7252     \ifglxtrinsertinside\else##2\fi
7253 }%
7254 \renewcommand*{\Glsxtrfullformat}[2]{%
7255     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7256     \ifglxtrinsertinside\else##2\fi
7257 }%
7258 \renewcommand*{\Glsxtrfullplformat}[2]{%
7259     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7260     \ifglxtrinsertinside\else##2\fi
7261 }%
7262 }

```

Set this as the default style for acronyms:

```

7263 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

7264 \letabbreviationstyle{short-nolong}{short}

```

short-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

7265 \newabbreviationstyle{short-nolong-noreg}%
7266 {%
7267     \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

7268 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7269     \glshasattribute{\the\glslabeltok}{regular}%
7270     {%
7271         \glissetattribute{\the\glslabeltok}{regular}{false}%
7272     }%
7273     {}%
7274 }%

```



```

7275 }%
7276 {%
7277   \GlsXtrUseAbbrStyleFmts{short-nolong}%
7278 }

```

trshortdescname

```

7279 \newcommand*{\glxtrshortdescname}{%
7280   \protect\glsabbrvfont{\the\glsshorttok}%
7281 }

```

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.

```

7282 \newabbreviationstyle{short-desc}%
7283 {%
7284   \renewcommand*{\CustomAbbreviationFields}{%
7285     name={\glxtrshortdescname},
7286     sort={\the\glsshorttok},
7287     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
7288     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
7289     text={\protect\glsabbrvfont{\the\glsshorttok}},
7290     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7291     description={\the\glslongtok}}%
7292   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7293     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7294 }%
7295 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7296 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7297 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7298 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7299 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7300 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7301 \renewcommand*{\glxtrinlinefullformat}[2]{%
7302   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7303   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7304   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7305 }%
7306 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7307   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7308   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7309   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7310 }%
7311 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7312   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7313   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7314   \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
7315 }%

```

```

7316 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7317   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7318   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7319   \glsxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7320 }%

```

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.

```

7321 \renewcommand*{\glsxtrfullformat}[2]{%
7322   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7323   \ifglxtrinsertinside\else##2\fi
7324 }%
7325 \renewcommand*{\glsxtrfullplformat}[2]{%
7326   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7327   \ifglxtrinsertinside\else##2\fi
7328 }%
7329 \renewcommand*{\Glsxtrfullformat}[2]{%
7330   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7331   \ifglxtrinsertinside\else##2\fi
7332 }%
7333 \renewcommand*{\Glsxtrfullplformat}[2]{%
7334   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7335   \ifglxtrinsertinside\else##2\fi
7336 }%
7337 }

```

ort-nolong-desc

```

7338 \letabbreviationstyle{short-nolong-desc}{short-desc}

```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

7339 \newabbreviationstyle{short-nolong-desc-noreg}%
7340 {%
7341   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%

```

Unset the regular attribute if it has been set.

```

7342 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7343   \glshasattribute{\the\glslabeltok}{regular}%
7344   {%
7345     \glssetattribute{\the\glslabeltok}{regular}{false}%
7346   }%
7347   {}%
7348 }%
7349 }%
7350 {%
7351   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
7352 }

```

nolong-short Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

7353 \newabbreviationstyle{nolong-short}%
7354 {%
7355   \GlsXtrUseAbbrStyleSetup{short-nolong}%
7356 }%
7357 {%
7358   \GlsXtrUseAbbrStyleFmts{short-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7359 \renewcommand*{\glxtrinlinefullformat}[2]{%
7360   \protect\glsfirstlongfont{\glsaccesslong{##1}}%
7361   \ifglxtrininsertinside##2\fi}%
7362 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7363 \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7364 }%
7365 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7366   \protect\glsfirstlongfont{\glsaccesslongpl{##1}}%
7367   \ifglxtrininsertinside##2\fi}%
7368 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7369 \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7370 }%
7371 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7372   \protect\glsfirstlongfont{\glsaccesslong{##1}}%
7373   \ifglxtrininsertinside##2\fi}%
7374 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7375 \glxtrparen{\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
7376 }%
7377 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7378   \protect\glsfirstlongfont{\glsaccesslongpl{##1}}%
7379   \ifglxtrininsertinside##2\fi}%
7380 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7381 \glxtrparen{\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
7382 }%
7383 }

```

`nolong-short-noreg` Like `nolong-short` but doesn't set the regular attribute.

```

7384 \newabbreviationstyle{nolong-short-noreg}%
7385 {%
7386   \GlsXtrUseAbbrStyleSetup{nolong-short}%

```

Unset the regular attribute if it has been set.

```

7387 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7388   \glshasattribute{\the\glslabeltok}{regular}%
7389   {%
7390     \glissetattribute{\the\glslabeltok}{regular}{false}%
7391   }%
7392   {}%
7393 }%
7394 }%
7395 {%
7396   \GlsXtrUseAbbrStyleFmts{nolong-short}%

```

7397 }

noshortdescname

```
7398 \newcommand*{\glxtrlongnoshortdescname}{%
7399   \protect\glslongfont{\the\glslongtok}%
7400 }
```

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.

```
7401 \newabbreviationstyle{long-desc}%
7402 {%
7403   \renewcommand*{\CustomAbbreviationFields}{%
7404     name={\glxtrlongnoshortdescname},
7405     sort={\the\glslongtok},
7406     first={\protect\glsfirstlongfont{\the\glslongtok}},
7407     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7408     text={\glslongfont{\the\glslongtok}},
7409     plural={\glslongfont{\the\glslongpltok}}}%
7410   }%
7411   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7412     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7413 }%
7414 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
7415 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7416 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7417 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7418 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7419 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
7420 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7421   \glslongfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7422   \ifglxtrininsertinside \else##2\fi
7423 }%
7424 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7425   \glslongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7426   \ifglxtrininsertinside \else##2\fi
7427 }%
7428 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7429   \glslongfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7430   \ifglxtrininsertinside \else##2\fi
7431 }%
7432 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7433   \glslongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7434   \ifglxtrininsertinside \else##2\fi
7435 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```

7436 \renewcommand*{\glxtrinlinefullformat}[2]{%
7437   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7438   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7439   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7440 }%
7441 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7442   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7443   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7444   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7445 }%
7446 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7447   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7448   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7449   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
7450 }%
7451 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7452   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7453   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7454   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
7455 }%

```

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.

```

7456 \renewcommand*{\glxtrfullformat}[2]{%
7457   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7458   \ifglxtrinsertinside\else##2\fi
7459 }%
7460 \renewcommand*{\glxtrfullplformat}[2]{%
7461   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7462   \ifglxtrinsertinside\else##2\fi
7463 }%
7464 \renewcommand*{\Glsxtrfullformat}[2]{%
7465   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7466   \ifglxtrinsertinside\else##2\fi
7467 }%
7468 \renewcommand*{\Glsxtrfullplformat}[2]{%
7469   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7470   \ifglxtrinsertinside\else##2\fi
7471 }%
7472 }

```

`ng-noshort-desc` Provide a synonym that matches similar styles.

```
7473 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

`short-desc-noreg` Like `long-noshort-desc` but doesn't set the regular attribute.

```

7474 \newabbreviationstyle{long-noshort-desc-noreg}%
7475 {%
7476   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%

```

Unset the regular attribute if it has been set.

```
7477 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7478   \glshasattribute{\the\glslabeltok}{regular}%
7479   {%
7480     \glsselattribute{\the\glslabeltok}{regular}{false}%
7481   }%
7482   {}%
7483 }%
7484 }%
7485 {%
7486   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
7487 }
```

longnoshortname

```
7488 \newcommand*{\glsxtrlongnoshortname}{%
7489   \protect\glsabbrvfont{\the\glsshorttok}%
7490 }
```

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.

```
7491 \newabbreviationstyle{long}%
7492 {%
7493   \renewcommand*{\CustomAbbreviationFields}{%
7494     name={\glsxtrlongnoshortname},
7495     sort={\the\glsshorttok},
7496     first={\protect\glsfirstlongfont{\the\glslongtok}},
7497     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7498     text={\glslongfont{\the\glslongtok}},
7499     plural={\glslongfont{\the\glslongpltok}},%
7500     description={\the\glslongtok}%
7501   }%
7502   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7503     \glsselattribute{\the\glslabeltok}{regular}{true}}%
7504 }%
7505 {%
7506   \GlsXtrUseAbbrStyleFmts{long-desc}%
7507 }
```

long-noshort Provide a synonym that matches similar styles.

```
7508 \letabbreviationstyle{long-noshort}{long}
```

g-noshort-noreg Like long-noshort but doesn't set the regular attribute.

```
7509 \newabbreviationstyle{long-noshort-noreg}%
7510 {%
7511   \GlsXtrUseAbbrStyleSetup{long-noshort}%
```

Unset the regular attribute if it has been set.

```
7512 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```

7513 \glshasattribute{\the\glslabeltok}{regular}%
7514 {%
7515 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
7516 }%
7517 {}%
7518 }%
7519 }%
7520 {%
7521 \GlsXtrUseAbbrStyleFmts{long-noshort}%
7522 }

```

1.7.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

`\glsxtrscfont` Maintained for backward-compatibility.

```
7523 \newcommand*{\glsxtrscfont}[1]{\textsc{#1}}
```

`\glssabbrvscfont` Added for consistent naming.

```
7524 \newcommand*{\glssabbrvscfont}{\glsxtrscfont}
```

`\glsxtrfirstscfont` Maintained for backward-compatibility.

```
7525 \newcommand*{\glsxtrfirstscfont}[1]{\glssabbrvscfont{#1}}
```

`\glssfirstabbrvscfont` Added for consistent naming.

```
7526 \newcommand*{\glssfirstabbrvscfont}{\glsxtrfirstscfont}
```

and for the default short form suffix:

`\glsxtrscsuffix`

```
7527 \newcommand*{\glsxtrscsuffix}{\glstextup{\glsxtrabbrvpluralsuffix}}
```

`long-short-sc`

```

7528 \newabbreviationstyle{long-short-sc}%
7529 {%
7530 \renewcommand*{\CustomAbbreviationFields}{%
7531   name={\glsxtrlongshortname},
7532   sort={\the\glssshorttok},
7533   first={\protect\glssfirstlongdefaultfont{\the\glslongtok}%
7534     \protect\glsxtrfullsep{\the\glslabeltok}%
7535     \glsxtrparen{\protect\glssfirstabbrvscfont{\the\glssshorttok}}},%
7536   firstplural={\protect\glssfirstlongdefaultfont{\the\glslongpltok}%
7537     \protect\glsxtrfullsep{\the\glslabeltok}%
7538     \glsxtrparen{\protect\glssfirstabbrvscfont{\the\glssshortpltok}}},%
7539   plural={\protect\glssabbrvscfont{\the\glssshortpltok}}},%
7540   description={\the\glslongtok}}%
7541 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7542 \glshasattribute{\the\glslabeltok}{regular}%
7543 }%

```

```

7544 \glsetattribute{\the\glslabeltok}{regular}{false}%
7545 }%
7546 {}%
7547 }%
7548 }%
7549 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7550 \renewcommand*\abbrvpluralsuffix{\protect\glxtrscsuffix}%
7551 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
7552 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%

```

Use the default long fonts.

```

7553 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7554 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7555 \renewcommand*\glsxtrfullformat[2]{%
7556 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7557 \ifglsxtrininsertinside\else##2\fi
7558 \glsxtrfullsep{##1}%
7559 \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7560 }%
7561 \renewcommand*\glsxtrfullplformat[2]{%
7562 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7563 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7564 \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7565 }%
7566 \renewcommand*\Glsxtrfullformat[2]{%
7567 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7568 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7569 \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7570 }%
7571 \renewcommand*\Glsxtrfullplformat[2]{%
7572 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7573 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7574 \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7575 }%
7576 }

```

g-short-sc-desc

```

7577 \newabbreviationstyle{long-short-sc-desc}%
7578 {%
7579 \renewcommand*\CustomAbbreviationFields{%
7580 name={\glsxtrlongshortdescname},
7581 sort={\glsxtrlongshortdescsort},%
7582 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7583 \protect\glsxtrfullsep{\the\glslabeltok}%
7584 \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
7585 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%

```



```

7586     \protect\glxtrfullsep{\the\glslabeltok}%
7587     \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
7588     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7589     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
7590 }%

```

Unset the regular attribute if it has been set.

```

7591 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7592   \glshasattribute{\the\glslabeltok}{regular}%
7593   {%
7594     \glissetattribute{\the\glslabeltok}{regular}{false}%
7595   }%
7596   {}%
7597 }%
7598 }%
7599 {%

```

As long-short-sc style:

```

7600 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
7601 }

```

Now the short (long) version

```

7602 \newabbreviationstyle{short-sc-long}%
7603 {%
7604   \renewcommand*{\CustomAbbreviationFields}{%
7605     name={\glxtrshortlongname},
7606     sort={\the\glsshorttok},
7607     description={\the\gslongtok},%
7608     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%
7609     \protect\glxtrfullsep{\the\glslabeltok}%
7610     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongtok}}},%
7611     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}}%
7612     \protect\glxtrfullsep{\the\glslabeltok}%
7613     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongpltok}}},%
7614     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7615 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7616   \glshasattribute{\the\glslabeltok}{regular}%
7617   {%
7618     \glissetattribute{\the\glslabeltok}{regular}{false}%
7619   }%
7620   {}%
7621 }%
7622 }%
7623 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7624 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7625 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
7626 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%

```

```

7627 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7628 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7629 \renewcommand*{\glsxtrfullformat}[2]{%
7630   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7631   \ifglsxtrininsertinside\else##2\fi
7632   \glsxtrfullsep{##1}%
7633   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7634 }%
7635 \renewcommand*{\glsxtrfullplformat}[2]{%
7636   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7637   \ifglsxtrininsertinside\else##2\fi
7638   \glsxtrfullsep{##1}%
7639   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7640 }%
7641 \renewcommand*{\Glsxtrfullformat}[2]{%
7642   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7643   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7644   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7645 }%
7646 \renewcommand*{\Glsxtrfullplformat}[2]{%
7647   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7648   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7649   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7650 }%
7651 }

```

As before but user provides description

```

7652 \newabbreviationstyle{short-sc-long-desc}%
7653 {%
7654   \renewcommand*{\CustomAbbreviationFields}{%
7655     name={\glsxtrshortlongdescname},
7656     sort={\glsxtrshortlongdescsort},
7657     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7658       \protect\glsxtrfullsep{\the\glslabeltok}%
7659       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7660     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7661       \protect\glsxtrfullsep{\the\glslabeltok}%
7662       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7663     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7664     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
7665   }%

```

Unset the regular attribute if it has been set.

```

7666 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7667   \glshasattribute{\the\glslabeltok}{regular}%
7668   {%
7669     \glissetattribute{\the\glslabeltok}{regular}{false}%
7670   }%

```

```

7671 {}%
7672 }%
7673 }%
7674 {%
    As short-sc-long style:
7675 \GlsXtrUseAbbrStyleFmts{short-sc-long}%
7676 }

```

short-sc

```

7677 \newabbreviationstyle{short-sc}%
7678 {%
7679 \renewcommand*{\CustomAbbreviationFields}{%
7680 name={\glxtrshortnolongname},
7681 sort={\the\glsshorttok},
7682 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7683 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7684 text={\protect\glsabbrvscfont{\the\glsshorttok}},
7685 plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7686 description={\the\glslongtok}}%
7687 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7688 \glsetattribute{\the\glslabeltok}{regular}{true}}%
7689 }%
7690 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7691 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7692 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7693 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7694 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7695 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7696 \renewcommand*{\glxtrinlinefullformat}[2]{%
7697 \protect\glsfirstabbrvscfont{\glsaccessshort{##1}}%
7698 \ifglxtrininsertinside##2\fi}%
7699 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7700 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7701 }%
7702 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7703 \protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}%
7704 \ifglxtrininsertinside##2\fi}%
7705 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7706 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7707 }%

7708 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7709 \protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}%
7710 \ifglxtrininsertinside##2\fi}%
7711 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7712 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%

```

```

7713 }%
7714 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7715   \protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}%
7716   \ifglxtrinsertinside##2\fi}%
7717   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7718   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7719 }%

```

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.

```

7720 \renewcommand*{\glxtrfullformat}[2]{%
7721   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7722   \ifglxtrinsertinside\else##2\fi
7723 }%
7724 \renewcommand*{\glxtrfullplformat}[2]{%
7725   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7726   \ifglxtrinsertinside\else##2\fi
7727 }%
7728 \renewcommand*{\Glsxtrfullformat}[2]{%
7729   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7730   \ifglxtrinsertinside\else##2\fi
7731 }%
7732 \renewcommand*{\Glsxtrfullplformat}[2]{%
7733   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7734   \ifglxtrinsertinside\else##2\fi
7735 }%
7736 }

```

short-sc-nolong

```

7737 \letabbreviationstyle{short-sc-nolong}{short-sc}

```

short-sc-desc

```

7738 \newabbreviationstyle{short-sc-desc}%
7739 {%
7740   \renewcommand*{\CustomAbbreviationFields}{%
7741     name={\glxtrshortdescname},
7742     sort={\the\glsshorttok},
7743     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7744     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7745     text={\protect\glsabbrvscfont{\the\glsshorttok}},
7746     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7747     description={\the\glslongtok}}%
7748   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7749     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7750 }%
7751 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7752 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7753 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%

```

```

7754 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
7755 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7756 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7757 \renewcommand*\glsxtrinlinefullformat}[2]{%
7758   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7759   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7760   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7761 }%
7762 \renewcommand*\glsxtrinlinefullplformat}[2]{%
7763   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7764   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7765   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7766 }%
7767 \renewcommand*\Glsxtrinlinefullformat}[2]{%
7768   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7769   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7770   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7771 }%
7772 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
7773   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7774   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7775   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7776 }%

```

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.

```

7777 \renewcommand*\glsxtrfullformat}[2]{%
7778   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7779   \ifglsxtrininsertinside\else##2\fi
7780 }%
7781 \renewcommand*\glsxtrfullplformat}[2]{%
7782   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7783   \ifglsxtrininsertinside\else##2\fi
7784 }%
7785 \renewcommand*\Glsxtrfullformat}[2]{%
7786   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7787   \ifglsxtrininsertinside\else##2\fi
7788 }%
7789 \renewcommand*\Glsxtrfullplformat}[2]{%
7790   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7791   \ifglsxtrininsertinside\else##2\fi
7792 }%
7793 }

```

-sc-nolong-desc

```

7794 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

```

nolong-short-sc

```

7795 \newabbreviationstyle{nolong-short-sc}%
7796 {%
7797   \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%
7798 }%
7799 {%
7800   \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7801 \renewcommand*{\glxtrinlinefullformat}[2]{%
7802   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}}%
7803   \ifglxtrinsertinside##2\fi}%
7804 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7805 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7806 }%
7807 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7808   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}%
7809   \ifglxtrinsertinside##2\fi}%
7810 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7811 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7812 }%
7813 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7814   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}}%
7815   \ifglxtrinsertinside##2\fi}%
7816 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7817 \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
7818 }%
7819 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7820   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}%
7821   \ifglxtrinsertinside##2\fi}%
7822 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7823 \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
7824 }%
7825 }

```

`long-noshort-sc` The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glxtrshort`.

```

7826 \newabbreviationstyle{long-noshort-sc}%
7827 {%
7828   \renewcommand*{\CustomAbbreviationFields}{%
7829     name={\glxtrlongnoshortname},
7830     sort={\the\glsshorttok},
7831     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7832     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
7833     text={\protect\glslongdefaultfont{\the\glslongtok}},
7834     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7835     description={\the\glslongtok}%
7836   }%
7837   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7838     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7839 }%

```

7840 {%

Use smallcaps and adjust the plural suffix to revert to upright.

```
7841 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7842 \renewcommand*{\glabbrvfont}[1]{\glabbrvscfont{##1}}%
7843 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvscfont{##1}}%
7844 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
7845 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
7846 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7847   \glslongdefaultfont{\glaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7848   \ifglxtrininsertinside \else##2\fi
7849 }%
7850 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7851   \glslongdefaultfont{\glaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7852   \ifglxtrininsertinside \else##2\fi
7853 }%
7854 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7855   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7856   \ifglxtrininsertinside \else##2\fi
7857 }%
7858 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7859   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7860   \ifglxtrininsertinside \else##2\fi
7861 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
7862 \renewcommand*{\glxtrinlinefullformat}[2]{%
7863   \glfirstlongdefaultfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7864   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7865   \glxtrparen{\protect\glfirstabbrvscfont{\glaccessshort{##1}}}%
7866 }%
7867 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7868   \glfirstlongdefaultfont{\glaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7869   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7870   \glxtrparen{\protect\glfirstabbrvscfont{\glaccessshortpl{##1}}}%
7871 }%
7872 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7873   \glfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7874   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7875   \glxtrparen{\protect\glfirstabbrvscfont{\glaccessshort{##1}}}%
7876 }%
7877 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7878   \glfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7879   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7880   \glxtrparen{\protect\glfirstabbrvscfont{\glaccessshortpl{##1}}}%
7881 }%
```

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.

```

7882 \renewcommand*{\glxtrfullformat}[2]{%
7883   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7884   \ifglxtrinsertinside\else##2\fi
7885 }%
7886 \renewcommand*{\glxtrfullplformat}[2]{%
7887   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7888   \ifglxtrinsertinside\else##2\fi
7889 }%
7890 \renewcommand*{\Glsxtrfullformat}[2]{%
7891   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7892   \ifglxtrinsertinside\else##2\fi
7893 }%
7894 \renewcommand*{\Glsxtrfullplformat}[2]{%
7895   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7896   \ifglxtrinsertinside\else##2\fi
7897 }%
7898 }

```

long-sc Backward compatibility:

```

7899 \@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.

```

7900 \newabbreviationstyle{long-noshort-sc-desc}%
7901 {%
7902   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7903 }%
7904 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7905 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7906 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7907 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7908 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7909 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7910 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7911   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7912   \ifglxtrinsertinside \else##2\fi
7913 }%
7914 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7915   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7916   \ifglxtrinsertinside \else##2\fi
7917 }%
7918 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7919   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7920   \ifglxtrinsertinside \else##2\fi
7921 }%
7922 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%

```



```

7923 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7924 \ifglxtrinsertinside \else##2\fi
7925 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7926 \renewcommand*{\glxtrinlinefullformat}[2]{%
7927 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7928 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7929 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7930 }%
7931 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7932 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7933 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7934 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7935 }%
7936 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7937 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7938 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7939 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7940 }%
7941 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7942 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7943 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7944 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7945 }%

```

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.

```

7946 \renewcommand*{\glxtrfullformat}[2]{%
7947 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7948 \ifglxtrinsertinside\else##2\fi
7949 }%
7950 \renewcommand*{\glxtrfullplformat}[2]{%
7951 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7952 \ifglxtrinsertinside\else##2\fi
7953 }%
7954 \renewcommand*{\Glsxtrfullformat}[2]{%
7955 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7956 \ifglxtrinsertinside\else##2\fi
7957 }%
7958 \renewcommand*{\Glsxtrfullplformat}[2]{%
7959 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7960 \ifglxtrinsertinside\else##2\fi
7961 }%
7962 }

```

long-desc-sc Backward compatibility:

```

7963 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

7964 \newabbreviationstyle{short-sc-footnote}%
7965 {%
7966   \renewcommand*{\CustomAbbreviationFields}{%
7967     name={\glxtrfootnotename},
7968     sort={\the\glsshorttok},
7969     description={\the\glslongtok},%
7970     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7971       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7972       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
7973     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7974       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7975       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
7976     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.

```

7977 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7978   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
7979   \glsattribute{\the\glslabeltok}{regular}%
7980   {%
7981     \glssetattribute{\the\glslabeltok}{regular}{false}%
7982   }%
7983   {}%
7984 }%
7985 }%
7986 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7987 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7988 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7989 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7990 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7991 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

7992 \renewcommand*{\glxtrfullformat}[2]{%
7993   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7994   \ifglxtrininsertinside\else##2\fi
7995   \protect\glxtrabbrvfootnote{##1}%
7996   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7997 }%
7998 \renewcommand*{\glxtrfullplformat}[2]{%
7999   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8000   \ifglxtrininsertinside\else##2\fi
8001   \protect\glxtrabbrvfootnote{##1}%
8002   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8003 }%
8004 \renewcommand*{\Glsxtrfullformat}[2]{%
8005   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8006   \ifglxtrininsertinside\else##2\fi
8007   \protect\glxtrabbrvfootnote{##1}%

```

```

8008     {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8009 }%
8010 \renewcommand*{\Glsxtrfullplformat}[2]{%
8011   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8012   \ifglxtrinsertinside\else##2\fi
8013   \protect\glxtrabbrvfootnote{##1}%
8014   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8015 }%

```

The first use full form and the inline full form use the short (long) style.

```

8016 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8017   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8018   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8019   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8020 }%
8021 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8022   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8023   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8024   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8025 }%
8026 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8027   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8028   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8029   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8030 }%
8031 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8032   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8033   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8034   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8035 }%
8036 }

```

footnote-sc Backward compatibility:

```

8037 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}

```

sc-postfootnote

```

8038 \newabbreviationstyle{short-sc-postfootnote}%
8039 {%
8040   \renewcommand*{\CustomAbbreviationFields}{%
8041     name={\glxtrfootnotename},
8042     sort={\the\glsshorttok},
8043     description={\the\glslongtok},%
8044     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
8045     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
8046     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

8047 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8048   \csdef{glxtrpostlink\glscategorylabel}{%

```

```

8049      \glxtrifwasfirstuse
8050      {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8051      \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
8052      {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
8053      }%
8054      {%
8055      }%
8056      \glshasattribute{\the\glslabeltok}{regular}%
8057      {%
8058      \glissetattribute{\the\glslabeltok}{regular}{false}%
8059      }%
8060      {%
8061      }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

8062  \renewcommand*{\glxtrsetupfulldefs}{%
8063  \let\glxtrifwasfirstuse\@secondoftwo
8064  }%
8065 }%
8066 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

8067  \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
8068  \renewcommand*{\glabbrvfont}[1]{\glabbrvscfont{##1}}%
8069  \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvscfont{##1}}%
8070  \renewcommand*{\glfirstlongfont}[1]{\glfirstlongfootnotefont{##1}}%
8071  \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8072  \renewcommand*{\glxtrfullformat}[2]{%
8073  \glfirstabbrvscfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8074  \ifglxtrininsertinside\else##2\fi
8075  }%
8076  \renewcommand*{\glxtrfullplformat}[2]{%
8077  \glfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8078  \ifglxtrininsertinside\else##2\fi
8079  }%
8080  \renewcommand*{\Glsxtrfullformat}[2]{%
8081  \glfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8082  \ifglxtrininsertinside\else##2\fi
8083  }%
8084  \renewcommand*{\Glsxtrfullplformat}[2]{%
8085  \glfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8086  \ifglxtrininsertinside\else##2\fi
8087  }%

```

The first use full form and the inline full form use the short (long) style.

```

8088 \renewcommand*{\glxtrinlinefullformat}[2]{%
8089   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8090   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8091   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8092 }%
8093 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8094   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8095   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8096   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8097 }%
8098 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8099   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8100   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8101   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8102 }%
8103 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8104   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8105   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8106   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8107 }%
8108 }

```

postfootnote-sc Backward compatibility:

```
8109 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}
```

1.7.4 Predefined Styles (Fake Small Capitals)

These styles require the `relese` package, which must be loaded by the user. These styles all use:

`\glxtrsmfont` Maintained for backward compatibility.

```
8110 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}
```

`\glsabbrvsmfont` Added for consistent naming.

```
8111 \newcommand*{\glsabbrvsmfont}{\glxtrsmfont}
```

`\glxtrfirstsmfont` Maintained for backward compatibility.

```
8112 \newcommand*{\glxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}
```

`\glfirstabbrvsmfont` Added for consistent naming.

```
8113 \newcommand*{\glfirstabbrvsmfont}{\glxtrfirstsmfont}
```

and for the default short form suffix:

`\glxtrsmsuffix`

```
8114 \newcommand*{\glxtrsmsuffix}{\glxtrabbrvpluralsuffix}
```

long-short-sm

```
8115 \newabbreviationstyle{long-short-sm}%
8116 {%
8117   \renewcommand*{\CustomAbbreviationFields}{%
8118     name={\glxtrlongshortname},
8119     sort={\the\glsshorttok},
8120     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8121       \protect\glxtrfullsep{\the\glslabeltok}%
8122       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
8123     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8124       \protect\glxtrfullsep{\the\glslabeltok}%
8125       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
8126     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%
8127     description={\the\glslongtok}}%
8128   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8129     \glshasattribute{\the\glslabeltok}{regular}%
8130     {%
8131       \glissetattribute{\the\glslabeltok}{regular}{false}%
8132     }%
8133   }%
8134 }%
8135 }%
8136 {%
8137   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8138   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8139   \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%

```

Use the default long fonts.

```
8140 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8141 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```
8142 \renewcommand*{\glxtrfullformat}[2]{%
8143   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8144   \ifglxtrininsertinside\else##2\fi
8145   \glxtrfullsep{##1}%
8146   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8147 }%
8148 \renewcommand*{\glxtrfullplformat}[2]{%
8149   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8150   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8151   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8152 }%
8153 \renewcommand*{\Glsxtrfullformat}[2]{%
8154   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8155   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8156   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8157 }%
8158 \renewcommand*{\Glsxtrfullplformat}[2]{%
8159   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

8160 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8161 \glxtrparen{\glsfirstabbrvsmfont{\glssaccessshortpl{##1}}}%
8162 }%
8163 }

```

g-short-sm-desc

```

8164 \newabbreviationstyle{long-short-sm-desc}%
8165 {%
8166 \renewcommand*{\CustomAbbreviationFields}{%
8167 name={\glxtrlongshortdescname},
8168 sort={\glxtrlongshortdescsort},%
8169 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8170 \protect\glxtrfullsep{\the\glslabeltok}%
8171 \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
8172 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8173 \protect\glxtrfullsep{\the\glslabeltok}%
8174 \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
8175 text={\protect\glssabbrvsmfont{\the\glsshorttok}},%
8176 plural={\protect\glssabbrvsmfont{\the\glsshortpltok}}}%
8177 }%

```

Unset the regular attribute if it has been set.

```

8178 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8179 \glshasattribute{\the\glslabeltok}{regular}%
8180 {%
8181 \glissetattribute{\the\glslabeltok}{regular}{false}%
8182 }%
8183 {}%
8184 }%
8185 }%
8186 {%

```

As long-short-sm style:

```

8187 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
8188 }

```

short-sm-long Now the short (long) version

```

8189 \newabbreviationstyle{short-sm-long}%
8190 {%
8191 \renewcommand*{\CustomAbbreviationFields}{%
8192 name={\glxtrshortlongname},
8193 sort={\the\glsshorttok},
8194 description={\the\glslongtok},%
8195 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8196 \protect\glxtrfullsep{\the\glslabeltok}%
8197 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8198 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8199 \protect\glxtrfullsep{\the\glslabeltok}%
8200 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8201 plural={\protect\glssabbrvsmfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8202 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8203   \glshasattribute{\the\glslabeltok}{regular}%
8204   {%
8205     \glsselattribute{\the\glslabeltok}{regular}{false}%
8206   }%
8207   {}%
8208 }%
8209 }%
8210 {%
8211 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8212 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8213 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxrtrmsuffix}%
8214 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8215 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8216 \renewcommand*{\glsxtrfullformat}[2]{%
8217   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8218   \ifglsxtrininsertinside\else##2\fi
8219   \glsxtrfullsep{##1}%
8220   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8221 }%
8222 \renewcommand*{\glsxtrfullplformat}[2]{%
8223   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8224   \ifglsxtrininsertinside\else##2\fi
8225   \glsxtrfullsep{##1}%
8226   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8227 }%
8228 \renewcommand*{\Glsxtrfullformat}[2]{%
8229   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8230   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8231   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8232 }%
8233 \renewcommand*{\Glsxtrfullplformat}[2]{%
8234   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8235   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8236   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8237 }%
8238 }

```

rt-sm-long-desc As before but user provides description

```

8239 \newabbreviationstyle{short-sm-long-desc}%
8240 {%
8241 \renewcommand*{\CustomAbbreviationFields}{%
8242   name={\glsxtrshortlongdescname},
8243   sort={\glsxtrshortlongdescsort},
8244   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}}%
8245   \protect\glsxtrfullsep{\the\glslabeltok}%

```



```

8246     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8247     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8248     \protect\glxtrfullsep{\the\glslabeltok}%
8249     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8250     text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
8251     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}%
8252 }%

```

Unset the regular attribute if it has been set.

```

8253 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8254     \glshasattribute{\the\glslabeltok}{regular}%
8255     {%
8256         \glssetattribute{\the\glslabeltok}{regular}{false}%
8257     }%
8258 }%
8259 }%
8260 }%
8261 {%

```

As short-sm-long style:

```

8262 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
8263 }

```

short-sm

```

8264 \newabbreviationstyle{short-sm}%
8265 {%
8266     \renewcommand*{\CustomAbbreviationFields}{%
8267         name={\glxtrshortnolongname},
8268         sort={\the\glsshorttok},
8269         first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8270         firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8271         text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8272         plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8273         description={\the\glslongtok}}%
8274     \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8275         \glssetattribute{\the\glslabeltok}{regular}{true}}%
8276 }%
8277 {%
8278     \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8279     \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8280     \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8281     \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8282     \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8283 \renewcommand*{\glxtrinlinefullformat}[2]{%
8284     \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
8285     \ifglxtrininsertinside##2\fi}%
8286     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8287     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%

```

```

8288 }%
8289 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8290   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}%
8291     \ifglxtrininsertinside##2\fi}%
8292   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8293   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8294 }%

8295 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8296   \protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}%
8297     \ifglxtrininsertinside##2\fi}%
8298   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8299   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8300 }%

8301 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8302   \protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}%
8303     \ifglxtrininsertinside##2\fi}%
8304   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8305   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8306 }%

```

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.

```

8307 \renewcommand*{\glxtrfullformat}[2]{%
8308   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8309   \ifglxtrininsertinside\else##2\fi
8310 }%

8311 \renewcommand*{\glxtrfullplformat}[2]{%
8312   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8313   \ifglxtrininsertinside\else##2\fi
8314 }%

8315 \renewcommand*{\Glsxtrfullformat}[2]{%
8316   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8317   \ifglxtrininsertinside\else##2\fi
8318 }%

8319 \renewcommand*{\Glsxtrfullplformat}[2]{%
8320   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8321   \ifglxtrininsertinside\else##2\fi
8322 }%
8323 }

```

short-sm-nolong

```
8324 \letabbreviationstyle{short-sm-nolong}{short-sm}
```

short-sm-desc

```

8325 \newabbreviationstyle{short-sm-desc}%
8326 {%
8327   \renewcommand*{\CustomAbbreviationFields}{%
8328     name={\glxtrshortdescname},

```

```

8329     sort={\the\glssshorttok},
8330     first={\protect\glsfirstabbrvsmfont{\the\glssshorttok}},
8331     firstplural={\protect\glsfirstabbrvsmfont{\the\glssshortpltok}},
8332     text={\protect\glsabbrvsmfont{\the\glssshorttok}},
8333     plural={\protect\glsabbrvsmfont{\the\glssshortpltok}},
8334     description={\the\glslongtok}}%
8335 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8336   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8337 }%
8338 {%
8339   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8340   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8341   \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%
8342   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8343   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8344   \renewcommand*\glsxtrinlinefullformat[2]{%
8345     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8346     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8347     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8348   }%
8349   \renewcommand*\glsxtrinlinefullplformat[2]{%
8350     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8351     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8352     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8353   }%
8354   \renewcommand*\Glsxtrinlinefullformat[2]{%
8355     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8356     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8357     \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
8358   }%
8359   \renewcommand*\Glsxtrinlinefullplformat[2]{%
8360     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8361     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8362     \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
8363   }%

```

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.

```

8364   \renewcommand*\glsxtrfullformat[2]{%
8365     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8366     \ifglsxtrininsertinside\else##2\fi
8367   }%
8368   \renewcommand*\glsxtrfullplformat[2]{%
8369     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8370     \ifglsxtrininsertinside\else##2\fi
8371   }%
8372   \renewcommand*\Glsxtrfullformat[2]{%
8373     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%

```

```

8374     \ifglxtrinsertinside\else##2\fi
8375 }%
8376 \renewcommand*{\Glsxtrfullplformat}[2]{%
8377     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8378     \ifglxtrinsertinside\else##2\fi
8379 }%
8380 }

```

-sm-nolong-desc

```

8381 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}

```

nolong-short-sm

```

8382 \newabbreviationstyle{nolong-short-sm}%
8383 {%
8384     \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
8385 }%
8386 {%
8387     \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8388 \renewcommand*{\glxtrinlinefullformat}[2]{%
8389     \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
8390         \ifglxtrinsertinside##2\fi}%
8391     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8392     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8393 }%
8394 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8395     \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8396         \ifglxtrinsertinside##2\fi}%
8397     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8398     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8399 }%
8400 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8401     \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8402         \ifglxtrinsertinside##2\fi}%
8403     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8404     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8405 }%
8406 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8407     \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8408         \ifglxtrinsertinside##2\fi}%
8409     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8410     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8411 }%
8412 }

```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

8413 \newabbreviationstyle{long-noshort-sm}%

```

```

8414 {%
8415   \renewcommand*{\CustomAbbreviationFields}{%
8416     name={\glxtrlongnosshortname},
8417     sort={\the\glssshorttok},
8418     first={\protect\glstfirstlongdefaultfont{\the\glslongtok}},
8419     firstplural={\protect\glstfirstlongdefaultfont{\the\glslongpltok}},
8420     text={\protect\glslongdefaultfont{\the\glslongtok}},
8421     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8422     description={\the\glslongtok}%
8423   }%
8424   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8425     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
8426 }%
8427 {%
8428   \renewcommand*{\glabbrvfont}[1]{\glabbrvsmfont{##1}}%
8429   \renewcommand*{\glstfirstabbrvfont}[1]{\glstfirstabbrvsmfont{##1}}%
8430   \renewcommand*{\abbrvpluralsuffix}{\protect\glstxrsmssuffix}%
8431   \renewcommand*{\glstfirstlongfont}[1]{\glstfirstlongdefaultfont{##1}}%
8432   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8433   \renewcommand*{\glstxrsubsequentfmt}[2]{%
8434     \glslongdefaultfont{\glssaccesslong{##1}\ifglstxrinsertinside ##2\fi}%
8435     \ifglstxrinsertinside \else##2\fi
8436   }%
8437   \renewcommand*{\glstxrsubsequentplfmt}[2]{%
8438     \glslongdefaultfont{\glssaccesslongpl{##1}\ifglstxrinsertinside ##2\fi}%
8439     \ifglstxrinsertinside \else##2\fi
8440   }%
8441   \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8442     \glslongdefaultfont{\Glsaccesslong{##1}\ifglstxrinsertinside ##2\fi}%
8443     \ifglstxrinsertinside \else##2\fi
8444   }%
8445   \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8446     \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglstxrinsertinside ##2\fi}%
8447     \ifglstxrinsertinside \else##2\fi
8448   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8449   \renewcommand*{\glstxrinlinefullformat}[2]{%
8450     \glstfirstlongdefaultfont{\glssaccesslong{##1}\ifglstxrinsertinside##2\fi}%
8451     \ifglstxrinsertinside\else##2\fi\glstxrfullsep{##1}%
8452     \glstxrparen{\protect\glstfirstabbrvsmfont{\glssaccessshort{##1}}}%
8453   }%
8454   \renewcommand*{\glstxrinlinefullplformat}[2]{%
8455     \glstfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglstxrinsertinside##2\fi}%
8456     \ifglstxrinsertinside\else##2\fi\glstxrfullsep{##1}%
8457     \glstxrparen{\protect\glstfirstabbrvsmfont{\glssaccessshortpl{##1}}}%
8458   }%
8459   \renewcommand*{\Glsxtrinlinefullformat}[2]{%

```

```

8460 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8461 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8462 \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8463 }%
8464 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8465 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8466 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8467 \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8468 }%

```

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.

```

8469 \renewcommand*{\glxtrfullformat}[2]{%
8470 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8471 \ifglxtrinsertinside\else##2\fi
8472 }%
8473 \renewcommand*{\glxtrfullplformat}[2]{%
8474 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8475 \ifglxtrinsertinside\else##2\fi
8476 }%
8477 \renewcommand*{\Glsxtrfullformat}[2]{%
8478 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8479 \ifglxtrinsertinside\else##2\fi
8480 }%
8481 \renewcommand*{\Glsxtrfullplformat}[2]{%
8482 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8483 \ifglxtrinsertinside\else##2\fi
8484 }%
8485 }

```

long-sm Backward compatibility:

```
8486 \@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`.

```

8487 \newabbreviationstyle{long-noshort-sm-desc}%
8488 {%
8489 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8490 }%
8491 {%
8492 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8493 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8494 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
8495 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8496 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8497 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8498 \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8499 \ifglxtrinsertinside \else##2\fi

```

```

8500 }%
8501 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8502   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8503   \ifglxtrinsertinside \else##2\fi
8504 }%
8505 \renewcommand*{\Glsxtrsubsequentffmt}[2]{%
8506   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8507   \ifglxtrinsertinside \else##2\fi
8508 }%
8509 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8510   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8511   \ifglxtrinsertinside \else##2\fi
8512 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8513 \renewcommand*{\glxtrinlinefullformat}[2]{%
8514   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8515   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8516   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8517 }%
8518 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8519   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8520   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8521   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8522 }%
8523 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8524   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8525   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8526   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8527 }%
8528 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8529   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8530   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8531   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8532 }%

```

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.

```

8533 \renewcommand*{\glxtrfullformat}[2]{%
8534   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8535   \ifglxtrinsertinside\else##2\fi
8536 }%
8537 \renewcommand*{\glxtrfullplformat}[2]{%
8538   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8539   \ifglxtrinsertinside\else##2\fi
8540 }%
8541 \renewcommand*{\Glsxtrfullformat}[2]{%
8542   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8543   \ifglxtrinsertinside\else##2\fi
8544 }%

```

```

8545 \renewcommand*{\Glsxtrfullplformat}[2]{%
8546   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8547   \ifglxtrinsertinside\else##2\fi
8548 }%
8549 }

```

long-desc-sm Backward compatibility:

```

8550 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```

8551 \newabbreviationstyle{short-sm-footnote}%
8552 {%
8553   \renewcommand*{\CustomAbbreviationFields}{%
8554     name={\glsxtrfootnotename},
8555     sort={\the\glsshorttok},
8556     description={\the\glslongtok},%
8557     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8558       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8559       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8560     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8561       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8562       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8563     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

8564 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8565   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8566   \glsattribute{\the\glslabeltok}{regular}%
8567   {%
8568     \glssetattribute{\the\glslabeltok}{regular}{false}%
8569   }%
8570   {}%
8571 }%
8572 }%
8573 {%
8574   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8575   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8576   \renewcommand*\abbrvpluralsuffix{\protect\glxtrrmsuffix}%
8577   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8578   \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

8579 \renewcommand*{\glsxtrfullformat}[2]{%
8580   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8581   \ifglxtrinsertinside\else##2\fi
8582   \protect\glxtrabbrvfootnote{##1}%
8583   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8584 }%
8585 \renewcommand*{\glsxtrfullplformat}[2]{%

```



```

8586 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8587 \ifglxtrinsertinside\else##2\fi
8588 \protect\glxtrabbrvfootnote{##1}%
8589 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8590 }%
8591 \renewcommand*{\Glsxtrfullformat}[2]{%
8592 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8593 \ifglxtrinsertinside\else##2\fi
8594 \protect\glxtrabbrvfootnote{##1}%
8595 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8596 }%
8597 \renewcommand*{\Glsxtrfullplformat}[2]{%
8598 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8599 \ifglxtrinsertinside\else##2\fi
8600 \protect\glxtrabbrvfootnote{##1}%
8601 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8602 }%

```

The first use full form and the inline full form use the short (long) style.

```

8603 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8604 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8605 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8606 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8607 }%
8608 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8609 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8610 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8611 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8612 }%
8613 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8614 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8615 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8616 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8617 }%
8618 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8619 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8620 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8621 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8622 }%
8623 }

```

footnote-sm Backward compatibility:

```

8624 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

sm-postfootnote

```

8625 \newabbreviationstyle{short-sm-postfootnote}%
8626 {%
8627 \renewcommand*{\CustomAbbreviationFields}{%
8628 name={\glxtrfootnotename},
8629 sort={\the\glsshorttok},

```

```

8630   description={\the\glslongtok},%
8631   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
8632   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
8633   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.

```

8634   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8635     \csdef{glsxtrpostlink\glscategorylabel}{%
8636       \glsxtrifwasfirstuse
8637       {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8638         \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}}%
8639         {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%
8640     }%
8641   }%
8642 }%
8643 \glshasattribute{\the\glslabeltok}{regular}%
8644 {%
8645   \glssetattribute{\the\glslabeltok}{regular}{false}%
8646 }%
8647 {}%
8648 }%

```

The footnote needs to be suppressed in the inline form, so `\glsxtrfull` must set the first use switch off.

```

8649   \renewcommand*{\glsxtrsetupfulldefs}{%
8650     \let\glsxtrifwasfirstuse\@secondoftwo
8651   }%
8652 }%
8653 {%
8654   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8655   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8656   \renewcommand*\{abbrvpluralsuffix}{\protect\glsxtrmsuffix}%
8657   \renewcommand*\{glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
8658   \renewcommand*\{glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8659   \renewcommand*\{glsxtrfullformat}[2]{%
8660     \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8661     \ifglsxtrininsertinside\else##2\fi
8662   }%
8663   \renewcommand*\{glsxtrfullplformat}[2]{%
8664     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8665     \ifglsxtrininsertinside\else##2\fi
8666   }%
8667   \renewcommand*\{Glsxtrfullformat}[2]{%
8668     \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8669     \ifglsxtrininsertinside\else##2\fi

```

```

8670 }%
8671 \renewcommand*{\Glsxtrfullplformat}[2]{%
8672   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8673   \ifglxtrinsertinside\else##2\fi
8674 }%

```

The first use full form and the inline full form use the short (long) style.

```

8675 \renewcommand*{\glxtrinlinefullformat}[2]{%
8676   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8677   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8678   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8679 }%
8680 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8681   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8682   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8683   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8684 }%
8685 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8686   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8687   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8688   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8689 }%
8690 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8691   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8692   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8693   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8694 }%
8695 }

```

postfootnote-sm Backward compatibility:

```

8696 \@glxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

1.7.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```

8697 \newcommand*{\glsabbrvemfont}[1]{\emph{##1}}%

```

`\glsfirstabbrvemfont`

```

8698 \newcommand*{\glsfirstabbrvemfont}[1]{\glsabbrvemfont{##1}}%

```

The default short form suffix:

`\glxtremsuffix`

```

8699 \newcommand*{\glxtremsuffix}{\glxtrabbrvpluralsuffix}

```

`\glsfirstlongemfont` Only used by the “long-em” styles.

```

8700 \newcommand*{\glsfirstlongemfont}[1]{\glslongemfont{##1}}%

```

`\glslongemfont` Only used by the “long-em” styles.

```
8701 \newcommand*{\glslongemfont}[1]{\emph{#1}}%
```

`long-short-em` The long form is just set in the default long font.

```
8702 \newabbreviationstyle{long-short-em}%
8703 {%
8704   \renewcommand*{\CustomAbbreviationFields}{%
8705     name={\glsxtrlongshortname},
8706     sort={\the\glsshorttok},
8707     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8708       \protect\glsxtrfullsep{\the\glslabeltok}%
8709       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8710     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8711       \protect\glsxtrfullsep{\the\glslabeltok}%
8712       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8713     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}},%
8714     description={\the\glslongtok}}%
8715   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8716     \glsattribute{\the\glslabeltok}{regular}%
8717     {%
8718       \glssetattribute{\the\glslabeltok}{regular}{false}%
8719     }}%
8720   {}%
8721 }%
8722 }%
8723 {%
8724   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8725   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8726   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%

```

Use the default long fonts.

```
8727 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8728 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8729 \renewcommand*{\glsxtrfullformat}[2]{%
8730   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8731   \ifglsxtrininsertinside\else##2\fi
8732   \glsxtrfullsep{##1}%
8733   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8734 }%
8735 \renewcommand*{\glsxtrfullplformat}[2]{%
8736   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8737   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8738   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8739 }%
8740 \renewcommand*{\Glsxtrfullformat}[2]{%
8741   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8742   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8743   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%

```

```

8744 }%
8745 \renewcommand*{\Glsxtrfullplformat}[2]{%
8746   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8747   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8748   \glsxtrparen{\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8749 }%
8750 }

```

g-short-em-desc

```

8751 \newabbreviationstyle{long-short-em-desc}%
8752 {%
8753   \renewcommand*{\CustomAbbreviationFields}{%
8754     name={\glsxtrlongshortdescname},
8755     sort={\glsxtrlongshortdescsort},%
8756     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8757       \protect\glsxtrfullsep{\the\glslabeltok}%
8758       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8759     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8760       \protect\glsxtrfullsep{\the\glslabeltok}%
8761       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8762     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8763     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
8764   }%

```

Unset the regular attribute if it has been set.

```

8765   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8766     \glshasattribute{\the\glslabeltok}{regular}%
8767     {%
8768       \glissetattribute{\the\glslabeltok}{regular}{false}%
8769     }%
8770   }%
8771 }%
8772 }%
8773 {%

```

As long-short-em style:

```

8774   \GlsXtrUseAbbrStyleFmts{long-short-em}%
8775 }

```

ong-em-short-em

```

8776 \newabbreviationstyle{long-em-short-em}%
8777 {%
8778   \glslongemfont is used in the description since \glsdesc doesn't set the style.
8779   \renewcommand*{\CustomAbbreviationFields}{%
8780     name={\glsxtrlongshortname},
8781     sort={\the\glsshorttok},
8782     first={\protect\glsfirstlongemfont{\the\glslongtok}%
8783       \protect\glsxtrfullsep{\the\glslabeltok}%
8784       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%

```

```

8784 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}}%
8785 \protect\glsextrfullsep{\the\glslabeltok}}%
8786 \glsextrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%

8787 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
8788 description={\protect\glsfirstlongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

8789 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8790 \glshasattribute{\the\glslabeltok}{regular}}%
8791 {%
8792 \glssetattribute{\the\glslabeltok}{regular}{false}}%
8793 }%
8794 {}%
8795 }%
8796 }%
8797 {%
8798 \renewcommand*{\abbrvpluralsuffix}{\protect\glsextremsuffix}%
8799 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8800 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8801 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8802 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8803 \renewcommand*{\glsextrfullformat}[2]{%
8804 \glsfirstlongemfont{\glsaccesslong{##1}\ifglsextrinsertinside##2\fi}%
8805 \ifglsextrinsertinside\else##2\fi
8806 \glsextrfullsep{##1}}%
8807 \glsextrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8808 }%
8809 \renewcommand*{\glsextrfullplformat}[2]{%
8810 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsextrinsertinside##2\fi}%
8811 \ifglsextrinsertinside\else##2\fi\glsextrfullsep{##1}}%
8812 \glsextrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8813 }%
8814 \renewcommand*{\GlsXtrfullformat}[2]{%
8815 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsextrinsertinside##2\fi}%
8816 \ifglsextrinsertinside\else##2\fi\glsextrfullsep{##1}}%
8817 \glsextrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8818 }%
8819 \renewcommand*{\GlsXtrfullplformat}[2]{%
8820 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsextrinsertinside##2\fi}%
8821 \ifglsextrinsertinside\else##2\fi\glsextrfullsep{##1}}%
8822 \glsextrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8823 }%
8824 }

```

m-short-em-desc

```

8825 \newabbreviationstyle{long-em-short-em-desc}%
8826 {%

```

```

8827 \renewcommand*{\CustomAbbreviationFields}{%
8828   name={\glxtrlongshortdescname},
8829   sort={\glxtrlongshortdescsort},%
8830   first={\protect\glsfirstlongemfont{\the\glslongtok}%
8831     \protect\glxtrfullsep{\the\glslabeltok}%
8832     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8833   firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
8834     \protect\glxtrfullsep{\the\glslabeltok}%
8835     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8836   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8837   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8838 }%

```

Unset the regular attribute if it has been set.

```

8839 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8840   \glshasattribute{\the\glslabeltok}{regular}%
8841   {%
8842     \glissetattribute{\the\glslabeltok}{regular}{false}%
8843   }%
8844   {}%
8845 }%
8846 }%
8847 {%
8848   \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
8849 }

```

short-em-long Now the short (long) version

```

8850 \newabbreviationstyle{short-em-long}%
8851 {%
8852   \renewcommand*{\CustomAbbreviationFields}{%
8853     name={\glxtrshortlongname},
8854     sort={\the\glsshorttok},
8855     description={\the\glslongtok},%
8856     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8857       \protect\glxtrfullsep{\the\glslabeltok}%
8858       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8859     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8860       \protect\glxtrfullsep{\the\glslabeltok}%
8861       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8862     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

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 {%

Mostly as short-long style:

```
8872 \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
8873 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8874 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8875 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8876 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8877 \renewcommand*{\glsxtrfullformat}[2]{%
8878   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8879   \ifglsxtrininsertinside\else##2\fi
8880   \glsxtrfullsep{##1}%
8881   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8882 }%
8883 \renewcommand*{\glsxtrfullplformat}[2]{%
8884   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8885   \ifglsxtrininsertinside\else##2\fi
8886   \glsxtrfullsep{##1}%
8887   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8888 }%
8889 \renewcommand*{\Glsxtrfullformat}[2]{%
8890   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8891   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8892   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8893 }%
8894 \renewcommand*{\Glsxtrfullplformat}[2]{%
8895   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8896   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8897   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8898 }%
8899 }
```

rt-em-long-desc As before but user provides description

```
8900 \newabbreviationstyle{short-em-long-desc}%
8901 {%
8902   \renewcommand*{\CustomAbbreviationFields}{%
8903     name={\glsxtrshortlongdescname},
8904     sort={\glsxtrshortlongdescsort},
8905     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
8906     \protect\glsxtrfullsep{\the\glslabeltok}%
8907     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8908     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
8909     \protect\glsxtrfullsep{\the\glslabeltok}%
8910     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8911     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8912     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
8913   }%
```

Unset the regular attribute if it has been set.


```

8914 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8915   \glshasattribute{\the\glslabeltok}{regular}%
8916   {%
8917     \glissetattribute{\the\glslabeltok}{regular}{false}%
8918   }%
8919   {}%
8920 }%
8921 }%
8922 {%
8923   \GlsXtrUseAbbrStyleFmts{short-em-long}%
8924 }

```

hort-em-long-em

```

8925 \newabbreviationstyle{short-em-long-em}%
8926 {%

```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```

8927 \renewcommand*{\CustomAbbreviationFields}{%
8928   name={\glsxtrshortlongname},
8929   sort={\the\glsshorttok},
8930   description={\protect\glslongemfont{\the\glslongtok}},%
8931   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8932     \protect\glsxtrfullsep{\the\glslabeltok}%
8933     \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8934   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8935     \protect\glsxtrfullsep{\the\glslabeltok}%
8936     \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8937   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}

```

Unset the regular attribute if it has been set.

```

8938 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8939   \glshasattribute{\the\glslabeltok}{regular}%
8940   {%
8941     \glissetattribute{\the\glslabeltok}{regular}{false}%
8942   }%
8943   {}%
8944 }%
8945 }%
8946 {%
8947   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsuffix}%
8948   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8949   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8950   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8951   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8952 \renewcommand*{\glsxtrfullformat}[2]{%
8953   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrinsetinside##2\fi}%
8954   \ifglsxtrinsetinside\else##2\fi
8955   \glsxtrfullsep{##1}%

```

```

8956 \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8957 }%
8958 \renewcommand*{\glsxtrfullplformat}[2]{%
8959 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8960 \ifglsxtrininsertinside\else##2\fi
8961 \glsxtrfullsep{##1}%
8962 \glsxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8963 }%
8964 \renewcommand*{\Glsxtrfullformat}[2]{%
8965 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8966 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8967 \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8968 }%
8969 \renewcommand*{\Glsxtrfullplformat}[2]{%
8970 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8971 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8972 \glsxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8973 }%
8974 }

```

em-long-em-desc

```

8975 \newabbreviationstyle{short-em-long-em-desc}%
8976 {%
8977 \renewcommand*{\CustomAbbreviationFields}{%
8978 name={\glsxtrshortlongdescname},%
8979 sort={\glsxtrshortlongdescsort},%
8980 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8981 \protect\glsxtrfullsep{\the\glslabeltok}%
8982 \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8983 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8984 \protect\glsxtrfullsep{\the\glslabeltok}%
8985 \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8986 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8987 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8988 }%

```

Unset the regular attribute if it has been set.

```

8989 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8990 \glshasattribute{\the\glslabeltok}{regular}%
8991 {%
8992 \glissetattribute{\the\glslabeltok}{regular}{false}%
8993 }%
8994 {}%
8995 }%
8996 }%
8997 {%
8998 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
8999 }

```

short-em

```

9000 \newabbreviationstyle{short-em}%
9001 {%
9002   \renewcommand*{\CustomAbbreviationFields}{%
9003     name={\glxtrshortnolongname},
9004     sort={\the\glsshorttok},
9005     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
9006     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
9007     text={\protect\glsabbrvemfont{\the\glsshorttok}},
9008     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
9009     description={\the\glslongtok}}%
9010   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9011     \glssetattribute{\the\glslabeltok}{regular}{true}}%
9012 }%
9013 {%
9014   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9015   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9016   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9017   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9018   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

9019   \renewcommand*{\glxtrinlinefullformat}[2]{%
9020     \protect\glsfirstabbrvemfont{\glsaccesssshort{##1}}%
9021     \ifglxtrininsertinside##2\fi}%
9022   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9023   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9024 }%
9025   \renewcommand*{\glxtrinlinefullplformat}[2]{%
9026     \protect\glsfirstabbrvemfont{\glsaccesssshortpl{##1}}%
9027     \ifglxtrininsertinside##2\fi}%
9028   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9029   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9030 }%
9031   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9032     \protect\glsfirstabbrvemfont{\Glsaccesssshort{##1}}%
9033     \ifglxtrininsertinside##2\fi}%
9034   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9035   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9036 }%
9037   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9038     \protect\glsfirstabbrvemfont{\Glsaccesssshortpl{##1}}%
9039     \ifglxtrininsertinside##2\fi}%
9040   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9041   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9042 }%

```

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.

```

9043   \renewcommand*{\glxtrfullformat}[2]{%

```

```

9044 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9045 \ifglxtrinsertinside\else##2\fi
9046 }%
9047 \renewcommand*{\glxtrfullplformat}[2]{%
9048 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9049 \ifglxtrinsertinside\else##2\fi
9050 }%
9051 \renewcommand*{\Glsxtrfullformat}[2]{%
9052 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9053 \ifglxtrinsertinside\else##2\fi
9054 }%
9055 \renewcommand*{\Glsxtrfullplformat}[2]{%
9056 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9057 \ifglxtrinsertinside\else##2\fi
9058 }%
9059 }

```

short-em-nolong

```

9060 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

9061 \newabbreviationstyle{short-em-desc}%
9062 {%
9063 \renewcommand*{\CustomAbbreviationFields}{%
9064 name={\glxtrshortdescname},
9065 sort={\the\glsshorttok},
9066 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
9067 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
9068 text={\protect\glsabbrvemfont{\the\glsshorttok}},
9069 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
9070 description={\the\glslongtok}}%
9071 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9072 \glssetattribute{\the\glslabeltok}{regular}{true}}%
9073 }%
9074 {%
9075 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9076 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9077 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9078 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9079 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

9080 \renewcommand*{\glxtrinlinefullformat}[2]{%
9081 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9082 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9083 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9084 }%
9085 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9086 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9087 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

9088 \glstrparen{\glfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9089 }%
9090 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9091 \glfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9092 \ifglxtrinsertinside\else##2\fi\glstrfullsep{##1}%
9093 \glstrparen{\glfirstlongdefaultfont{\glsaccesslong{##1}}}%
9094 }%
9095 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9096 \glfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9097 \ifglxtrinsertinside\else##2\fi\glstrfullsep{##1}%
9098 \glstrparen{\glfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9099 }%

```

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.

```

9100 \renewcommand*{\glstrfullformat}[2]{%
9101 \glfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9102 \ifglxtrinsertinside\else##2\fi
9103 }%
9104 \renewcommand*{\glstrfullplformat}[2]{%
9105 \glfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9106 \ifglxtrinsertinside\else##2\fi
9107 }%
9108 \renewcommand*{\Glsxtrfullformat}[2]{%
9109 \glfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9110 \ifglxtrinsertinside\else##2\fi
9111 }%
9112 \renewcommand*{\Glsxtrfullplformat}[2]{%
9113 \glfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9114 \ifglxtrinsertinside\else##2\fi
9115 }%
9116 }

```

-em-nolong-desc

```

9117 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

9118 \newabbreviationstyle{nolong-short-em}%
9119 {%
9120 \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
9121 }%
9122 {%
9123 \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

9124 \renewcommand*{\glxtrinlinefullformat}[2]{%
9125 \protect\glfirstlongdefaultfont{\glsaccesslong{##1}%
9126 \ifglxtrinsertinside##2\fi}%
9127 \ifglxtrinsertinside\else##2\fi\glstrfullsep{##1}%
9128 \glstrparen{\glfirstabbrvemfont{\glsaccessshort{##1}}}%

```

```

9129 }%
9130 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9131   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
9132     \ifglxtrininsertinside##2\fi}%
9133   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9134   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9135 }%
9136 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9137   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
9138     \ifglxtrininsertinside##2\fi}%
9139   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9140   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9141 }%
9142 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9143   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
9144     \ifglxtrininsertinside##2\fi}%
9145   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9146   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9147 }%
9148 }

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

9149 \newabbreviationstyle{long-noshort-em}%
9150 {%
9151   \renewcommand*{\CustomAbbreviationFields}{%
9152     name={\glxtrlongnoshortname},
9153     sort={\the\glsshorttok},
9154     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
9155     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
9156     text={\protect\glslongdefaultfont{\the\glslongtok}},
9157     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
9158     description={\the\glslongtok}%
9159   }%
9160   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9161     \glssetattribute{\the\glslabeltok}{regular}{true}}%
9162 }%
9163 {%
9164   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9165   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9166   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9167   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9168   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9169 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9170   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9171   \ifglxtrininsertinside \else##2\fi
9172 }%
9173 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9174   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%

```

```

9175 \ifglxtrinsertinside \else##2\fi
9176 }%
9177 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9178 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9179 \ifglxtrinsertinside \else##2\fi
9180 }%
9181 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9182 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9183 \ifglxtrinsertinside \else##2\fi
9184 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9185 \renewcommand*{\glxtrinlinefullformat}[2]{%
9186 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9187 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9188 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
9189 }%
9190 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9191 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9192 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9193 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
9194 }%
9195 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9196 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9197 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9198 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
9199 }%
9200 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9201 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9202 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9203 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
9204 }%

```

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.

```

9205 \renewcommand*{\glxtrfullformat}[2]{%
9206 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9207 \ifglxtrinsertinside\else##2\fi
9208 }%
9209 \renewcommand*{\glxtrfullplformat}[2]{%
9210 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9211 \ifglxtrinsertinside\else##2\fi
9212 }%
9213 \renewcommand*{\Glsxtrfullformat}[2]{%
9214 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9215 \ifglxtrinsertinside\else##2\fi
9216 }%
9217 \renewcommand*{\Glsxtrfullplformat}[2]{%
9218 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9219 \ifglxtrinsertinside\else##2\fi

```

```

9220 }%
9221 }

```

long-em Backward compatibility:

```

9222 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}

```

g-em-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

9223 \newabbreviationstyle{long-em-noshort-em}%
9224 {%
9225   \renewcommand*{\CustomAbbreviationFields}{%
9226     name={\glxtrlongnoshortname},
9227     sort={\the\glsshorttok},
9228     first={\protect\glslongemfont{\the\glslongtok}},
9229     firstplural={\protect\glslongemfont{\the\glslongpltok}},
9230     text={\protect\glslongemfont{\the\glslongtok}},
9231     plural={\protect\glslongemfont{\the\glslongpltok}},%
9232     description={\protect\glslongemfont{\the\glslongtok}}%
9233   }%
9234   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9235     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9236 }%
9237 {%
9238   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9239   \renewcommand*{\glabbrvfont}[1]{\glabbrvemfont{##1}}%
9240   \renewcommand*{\glslfirstabbrvfont}[1]{\glslfirstabbrvemfont{##1}}%
9241   \renewcommand*{\glslfirstlongfont}[1]{\glslfirstlongemfont{##1}}%
9242   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9243 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9244   \glslongemfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9245   \ifglxtrininsertinside \else##2\fi
9246 }%
9247 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9248   \glslongemfont{\glssaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9249   \ifglxtrininsertinside \else##2\fi
9250 }%
9251 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9252   \glslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9253   \ifglxtrininsertinside \else##2\fi
9254 }%
9255 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9256   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9257   \ifglxtrininsertinside \else##2\fi
9258 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9259 \renewcommand*{\glxtrinlinefullformat}[2]{%
9260   \glslfirstlongemfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9261   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```



```

9262 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9263 }%
9264 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9265 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9266 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9267 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9268 }%
9269 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9270 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9271 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9272 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9273 }%
9274 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9275 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9276 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9277 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9278 }%

```

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.

```

9279 \renewcommand*{\glsxtrfullformat}[2]{%
9280 \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9281 \ifglsxtrinsertinside\else##2\fi
9282 }%
9283 \renewcommand*{\glsxtrfullplformat}[2]{%
9284 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9285 \ifglsxtrinsertinside\else##2\fi
9286 }%
9287 \renewcommand*{\Glsxtrfullformat}[2]{%
9288 \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9289 \ifglsxtrinsertinside\else##2\fi
9290 }%
9291 \renewcommand*{\Glsxtrfullplformat}[2]{%
9292 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9293 \ifglsxtrinsertinside\else##2\fi
9294 }%
9295 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

9296 \newabbreviationstyle{long-em-noshort-em-noreg}%
9297 {%
9298 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

9299 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9300 \glshasattribute{\the\glslabeltok}{regular}%
9301 {%
9302 \glssetattribute{\the\glslabeltok}{regular}{false}%
9303 }%
9304 }%

```

```

9305 }%
9306 }%
9307 {%
9308   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
9309 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glssshort`.

```

9310 \newabbreviationstyle{long-noshort-em-desc}%
9311 {%
9312   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9313 }%
9314 {%
9315   \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
9316   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
9317   \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvemfont{##1}}%
9318   \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
9319   \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9320 \renewcommand*{\glssxtrsubsequentfmt}[2]{%
9321   \glsslongdefaultfont{\glssaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
9322   \ifglssxtrininsertinside \else##2\fi
9323 }%
9324 \renewcommand*{\glssxtrsubsequentplfmt}[2]{%
9325   \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
9326   \ifglssxtrininsertinside \else##2\fi
9327 }%
9328 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9329   \glsslongdefaultfont{\Glsaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
9330   \ifglssxtrininsertinside \else##2\fi
9331 }%
9332 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9333   \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
9334   \ifglssxtrininsertinside \else##2\fi
9335 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9336 \renewcommand*{\glssxtrinlinefullformat}[2]{%
9337   \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
9338   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9339   \glssxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshort{##1}}}%
9340 }%
9341 \renewcommand*{\glssxtrinlinefullplformat}[2]{%
9342   \glssfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
9343   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9344   \glssxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9345 }%
9346 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9347   \glssfirstlongdefaultfont{\Glsaccesslong{##1}\ifglssxtrininsertinside##2\fi}%

```

```

9348     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9349     \glxtrparen{\protect\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
9350 }%
9351 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9352     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9353     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9354     \glxtrparen{\protect\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9355 }%

```

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.

```

9356 \renewcommand*{\glxtrfullformat}[2]{%
9357     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9358     \ifglxtrinsertinside\else##2\fi
9359 }%
9360 \renewcommand*{\glxtrfullplformat}[2]{%
9361     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9362     \ifglxtrinsertinside\else##2\fi
9363 }%
9364 \renewcommand*{\Glsxtrfullformat}[2]{%
9365     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9366     \ifglxtrinsertinside\else##2\fi
9367 }%
9368 \renewcommand*{\Glsxtrfullplformat}[2]{%
9369     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9370     \ifglxtrinsertinside\else##2\fi
9371 }%
9372 }

```

long-desc-em Backward compatibility:

```

9373 \@glxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}

```

noshort-em-desc The short form is explicitly invoked through commands like `\glssshort`. The long form is emphasized.

```

9374 \newabbreviationstyle{long-em-noshort-em-desc}%
9375 {%
9376 \renewcommand*{\CustomAbbreviationFields}{%
9377     name={\glxtrlongnoshortdescname},
9378     sort={\the\glslongtok},
9379     first={\protect\glsfirstlongemfont{\the\glslongtok}},
9380     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
9381     text={\glslongemfont{\the\glslongtok}},
9382     plural={\glslongemfont{\the\glslongpltok}}%
9383 }%
9384 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9385     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9386 }%
9387 {%
9388 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%

```

```

9389 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
9390 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
9391 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
9392 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9393 \renewcommand*\glsxtrsubsequentfmt[2]{%
9394   \glslongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9395   \ifglsxtrininsertinside \else##2\fi
9396 }%
9397 \renewcommand*\glsxtrsubsequentplfmt[2]{%
9398   \glslongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9399   \ifglsxtrininsertinside \else##2\fi
9400 }%
9401 \renewcommand*\Glsxtrsubsequentfmt[2]{%
9402   \glslongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9403   \ifglsxtrininsertinside \else##2\fi
9404 }%
9405 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
9406   \glslongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9407   \ifglsxtrininsertinside \else##2\fi
9408 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9409 \renewcommand*\glsxtrinlinefullformat[2]{%
9410   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9411   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9412   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9413 }%
9414 \renewcommand*\glsxtrinlinefullplformat[2]{%
9415   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9416   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9417   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9418 }%
9419 \renewcommand*\Glsxtrinlinefullformat[2]{%
9420   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9421   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9422   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9423 }%
9424 \renewcommand*\Glsxtrinlinefullplformat[2]{%
9425   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9426   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9427   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9428 }%

```

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.

```

9429 \renewcommand*\glsxtrfullformat[2]{%
9430   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9431   \ifglsxtrininsertinside\else##2\fi
9432 }%

```

```

9433 \renewcommand*{\glxtrfullplformat}[2]{%
9434   \glsfirstlongemfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9435   \ifglxtrininsertinside\else##2\fi
9436 }%
9437 \renewcommand*{\Glsxtrfullformat}[2]{%
9438   \glsfirstlongemfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9439   \ifglxtrininsertinside\else##2\fi
9440 }%
9441 \renewcommand*{\Glsxtrfullplformat}[2]{%
9442   \glsfirstlongemfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9443   \ifglxtrininsertinside\else##2\fi
9444 }%
9445 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

9446 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
9447 {%
9448   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%
   Unset the regular attribute if it has been set.
9449   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9450     \glshasattribute{\the\glslabeltok}{regular}%
9451     {%
9452       \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9453     }%
9454   }%
9455 }%
9456 }%
9457 {%
9458   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
9459 }

```

ort-em-footnote

```

9460 \newabbreviationstyle{short-em-footnote}%
9461 {%
9462   \renewcommand*{\CustomAbbreviationFields}{%
9463     name={\glxtrfootnotename},
9464     sort={\the\glssshorttok},
9465     description={\the\glslongtok},%
9466     first={\protect\glsfirstabbrvemfont{\the\glssshorttok}%
9467       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9468       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9469     firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}%
9470       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9471       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9472     plural={\protect\glssabbrvemfont{\the\glssshortpltok}}}%
   Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute
   if it has been set.
9473   \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

9474 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9475 \glsattribute{\the\glslabeltok}{regular}%
9476 {%
9477 \glssetattribute{\the\glslabeltok}{regular}{false}%
9478 }%
9479 {}%
9480 }%
9481 }%
9482 {%
9483 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
9484 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9485 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9486 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9487 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9488 \renewcommand*{\glsxtrfullformat}[2]{%
9489 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9490 \ifglsxtrininsertinside\else##2\fi
9491 \protect\glsxtrabbrvfootnote{##1}%
9492 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9493 }%
9494 \renewcommand*{\glsxtrfullplformat}[2]{%
9495 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9496 \ifglsxtrininsertinside\else##2\fi
9497 \protect\glsxtrabbrvfootnote{##1}%
9498 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9499 }%
9500 \renewcommand*{\Glsxtrfullformat}[2]{%
9501 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9502 \ifglsxtrininsertinside\else##2\fi
9503 \protect\glsxtrabbrvfootnote{##1}%
9504 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9505 }%
9506 \renewcommand*{\Glsxtrfullplformat}[2]{%
9507 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9508 \ifglsxtrininsertinside\else##2\fi
9509 \protect\glsxtrabbrvfootnote{##1}%
9510 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9511 }%

```

The first use full form and the inline full form use the short (long) style.

```

9512 \renewcommand*{\glsxtrininlinefullformat}[2]{%
9513 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9514 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9515 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9516 }%
9517 \renewcommand*{\glsxtrininlinefullplformat}[2]{%
9518 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9519 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%

```

```

9520 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9521 }%
9522 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9523 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9524 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9525 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9526 }%
9527 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9528 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9529 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9530 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9531 }%
9532 }

```

footnote-em Backward compatibility:

```

9533 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

em-postfootnote

```

9534 \newabbreviationstyle{short-em-postfootnote}%
9535 {%
9536 \renewcommand*{\CustomAbbreviationFields}{%
9537 name={\glxtrfootnotename},
9538 sort={\the\glsshorttok},
9539 description={\the\glslongtok},%
9540 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},%
9541 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},%
9542 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

9543 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9544 \csdef{glxtrpostlink\glscategorylabel}{%
9545 \glxtrifwasfirstuse
9546 }%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

9547 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9548 {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
9549 }%
9550 {}%
9551 }%
9552 \glshasattribute{\the\glslabeltok}{regular}%
9553 {%
9554 \glissetattribute{\the\glslabeltok}{regular}{false}%
9555 }%
9556 {}%
9557 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

9558 \renewcommand*\glxtrsetupfulldefs}{%
9559 \let\glxtrifwasfirstuse\@secondoftwo
9560 }%
9561 }%
9562 {%
9563 \renewcommand*\abbrvpluralsuffix{\protect\glxtremsuffix}%
9564 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
9565 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
9566 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
9567 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

9568 \renewcommand*\glxtrfullformat}[2]{%
9569 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9570 \ifglxtrininsertinside\else##2\fi
9571 }%
9572 \renewcommand*\glxtrfullplformat}[2]{%
9573 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9574 \ifglxtrininsertinside\else##2\fi
9575 }%
9576 \renewcommand*\Glsxtrfullformat}[2]{%
9577 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9578 \ifglxtrininsertinside\else##2\fi
9579 }%
9580 \renewcommand*\Glsxtrfullplformat}[2]{%
9581 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9582 \ifglxtrininsertinside\else##2\fi
9583 }%

```

The first use full form and the inline full form use the short (long) style.

```

9584 \renewcommand*\glxtrininlinefullformat}[2]{%
9585 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9586 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9587 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9588 }%
9589 \renewcommand*\glxtrininlinefullplformat}[2]{%
9590 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9591 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9592 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9593 }%
9594 \renewcommand*\Glsxtrininlinefullformat}[2]{%
9595 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9596 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9597 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
9598 }%
9599 \renewcommand*\Glsxtrininlinefullplformat}[2]{%
9600 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9601 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```



```

9602     \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9603   }%
9604 }

```

postfootnote-em Backward compatibility:

```

9605 \@glsxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}

```

1.7.6 Predefined Styles (User Parentheses Hook)

These styles allow the user to adjust the parenthetical forms. These styles all test for the existence of the field given by:

glsxtruserfield Default is the useri field.

```

9606 \newcommand*{\glsxtruserfield}{useri}

```

glsxtruserparen The format of the parenthetical information. The first argument is the long/short form. The second argument is the entry's label. If \glscurrentfieldvalue has been defined, then we have at least glossaries v4.23, which makes it easier for the user to adjust this.

```

9607 \ifdef\glscurrentfieldvalue
9608 {
9609   \newcommand*{\glsxtruserparen}[2]{%
9610     \glsxtrfullsep{#2}%
9611     \glsxtrparen
9612       {#1\ifglshasfield{\glsxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
9613   }
9614 }
9615 {
9616   \newcommand*{\glsxtruserparen}[2]{%
9617     \glsxtrfullsep{#2}%
9618     \glsxtrparen
9619       {#1\ifglshasfield{\glsxtruserfield}{#2}{, \@glo@thisvalue}{}}%
9620   }
9621 }

```

Font used for short form:

lsabbrvuserfont

```

9622 \newcommand*{\glsabbrvuserfont}[1]{\glsabbrvdefaultfont{#1}}

```

Font used for short form on first use:

stabbrvuserfont

```

9623 \newcommand*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}

```

Font used for long form:

glslonguserfont

```

9624 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}

```

Font used for long form on first use:

rstlonguserfont

```
9625 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
9626 \newcommand*{\glsxtrusersuffix}{\glsxtrabbrvpluralsuffix}
```

Description encapsulator.

userdescription The first argument is the description. The second argument is the label.

```
9627 \newcommand*{\glsuserdescription}[2]{\glslonguserfont{#1}}
```

long-short-user

```
9628 \newabbreviationstyle{long-short-user}%
9629 {%
9630   \renewcommand*{\CustomAbbreviationFields}{%
9631     name={\glsxtrlongshortname},
9632     sort={\the\glsshorttok},
9633     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9634       \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9635     {\the\glslabeltok}},%
9636     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9637       \protect\glsxtruserparen
9638       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9639     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9640     description={\protect\glsuserdescription{\the\glslongtok}%
9641       {\the\glslabeltok}}}%
9641 }
```

Unset the regular attribute if it has been set.

```
9642 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9643   \glshasattribute{\the\glslabeltok}{regular}%
9644   {%
9645     \glissetattribute{\the\glslabeltok}{regular}{false}%
9646   }%
9647   {}%
9648 }%
9649 }%
9650 }
```

In case the user wants to mix and match font styles, these are redefined here.

```
9651 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9652 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9653 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9654 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9655 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9656 \renewcommand*{\glsxtrfullformat}[2]{%
9657   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglsxtrinsetinside##2\fi}}%
```

```

9658     \ifglxtrinsertinside\else##2\fi
9659     \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9660 }%
9661 \renewcommand*{\glxtrfullplformat}[2]{%
9662     \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9663     \ifglxtrinsertinside\else##2\fi
9664     \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9665 }%
9666 \renewcommand*{\Glsxtrfullformat}[2]{%
9667     \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9668     \ifglxtrinsertinside\else##2\fi
9669     \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9670 }%
9671 \renewcommand*{\Glsxtrfullplformat}[2]{%
9672     \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9673     \ifglxtrinsertinside\else##2\fi
9674     \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9675 }%
9676 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

9677 \newabbreviationstyle{long-postshort-user}%
9678 {%
9679     \renewcommand*{\CustomAbbreviationFields}{%
9680         name={\glxtrlongshortname},
9681         sort={\the\glssshorttok},
9682         first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9683         firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9684         plural={\protect\glssabbrvuserfont{\the\glssshortpltok}},%
9685         description={\protect\glssuserdescription{\the\glslongtok}%
9686             {\the\glslabeltok}}}%
9687     \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9688         \csdef{glxtrpostlink\glscategorylabel}{%
9689             \glxtrifwasfirstuse
9690             {%
9691                 \glxtruserparen
9692                 {\glsfirstabbrvuserfont{\glssentryshort{\glslabel}}}%
9693                 {\glslabel}%
9694             }%
9695             {}%
9696         }%
9697         \glshasattribute{\the\glslabeltok}{regular}%
9698         {%
9699             \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9700         }%
9701         {}%
9702     }%
9703 }%
9704 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
9705 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9706 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9707 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9708 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9709 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

First use full form:

```
9710 \renewcommand*{\glxtrfullformat}[2]{%
9711   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9712   \ifglxtrininsertinside\else##2\fi
9713 }%
9714 \renewcommand*{\glxtrfullplformat}[2]{%
9715   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9716   \ifglxtrininsertinside\else##2\fi
9717 }%
9718 \renewcommand*{\Glsxtrfullformat}[2]{%
9719   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9720   \ifglxtrininsertinside\else##2\fi
9721 }%
9722 \renewcommand*{\Glsxtrfullplformat}[2]{%
9723   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9724   \ifglxtrininsertinside\else##2\fi
9725 }%
```

In-line format:

```
9726 \renewcommand*{\glxtrinlinefullformat}[2]{%
9727   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9728   \ifglxtrininsertinside\else##2\fi
9729   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9730 }%
9731 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9732   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9733   \ifglxtrininsertinside\else##2\fi
9734   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9735 }%
9736 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9737   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9738   \ifglxtrininsertinside\else##2\fi
9739   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9740 }%
9741 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9742   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9743   \ifglxtrininsertinside\else##2\fi
9744   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9745 }%
9746 }
```

ortuserdesname

```
9747 \newcommand*{\glxtrlongshortuserdesname}{%
```

```

9748 \protect\glslonguserfont{\the\glslongtok}%
9749 \protect\glsxtruserparen
9750 {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}%
9751 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

9752 \newabbreviationstyle{long-postshort-user-desc}%
9753 {%
9754   \renewcommand*{\CustomAbbreviationFields}{%
9755     name={\glsxtrlongshortuserdescname},
9756     sort={\the\glslongtok},
9757     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9758     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9759     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9760     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
9761   }%
9762   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9763     \csdef{glsxtrpostlink\glscategorylabel}{%
9764       \glsxtrifwasfirstuse
9765       {%
9766         \glsxtruserparen
9767         {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
9768         {\glslabel}%
9769       }%
9770     }%
9771   }%
9772   \glsattribute{\the\glslabeltok}{regular}%
9773   {%
9774     \glssetattribute{\the\glslabeltok}{regular}{false}%
9775   }%
9776   {}%
9777 }%
9778 }%
9779 {%
9780   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
9781 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

9782 \newabbreviationstyle{short-postlong-user}%
9783 {%
9784   \renewcommand*{\CustomAbbreviationFields}{%
9785     name={\glsxtrshortlongname},
9786     sort={\the\glsshorttok},
9787     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9788     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9789     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9790     description={\protect\glsuserdescription{\the\glslongtok}}%
9791     {\the\glslabeltok}}}%

```

```

9792 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9793   \csdef{glxtrpostlink\glscategorylabel}{%
9794     \glxtrifwasfirstuse
9795     {%
9796       \glxtruserparen
9797       {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9798       {\glslabel}%
9799     }%
9800   }%
9801 }%
9802 \glshasattribute{\the\glslabeltok}{regular}%
9803 {%
9804   \glissetattribute{\the\glslabeltok}{regular}{false}%
9805 }%
9806 {}%
9807 }%
9808 }%
9809 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9810 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9811 \renewcommand*{\glssabrvfont}[1]{\glssabrvuserfont{##1}}%
9812 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9813 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9814 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9815 \renewcommand*{\glxtrfullformat}[2]{%
9816   \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9817   \ifglxtrininsertinside\else##2\fi
9818 }%
9819 \renewcommand*{\glxtrfullplformat}[2]{%
9820   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9821   \ifglxtrininsertinside\else##2\fi
9822 }%
9823 \renewcommand*{\Glsxtrfullformat}[2]{%
9824   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9825   \ifglxtrininsertinside\else##2\fi
9826 }%
9827 \renewcommand*{\Glsxtrfullplformat}[2]{%
9828   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9829   \ifglxtrininsertinside\else##2\fi
9830 }%

```

In-line format:

```

9831 \renewcommand*{\glxtrininlinefullformat}[2]{%
9832   \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9833   \ifglxtrininsertinside\else##2\fi
9834   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9835 }%

```

```

9836 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9837   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9838   \ifglxtrinsertinside\else##2\fi
9839   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9840 }%
9841 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9842   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9843   \ifglxtrinsertinside\else##2\fi
9844   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9845 }%
9846 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9847   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9848   \ifglxtrinsertinside\else##2\fi
9849   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9850 }%
9851 }

```

onguserdescname

```

9852 \newcommand*{\glxtrshortlonguserdescname}{%
9853   \protect\glssabbrvuserfont{\the\glssshorttok}%
9854   \protect\glxtruserparen
9855     {\protect\glslonguserfont{\the\glslongpltok}}}%
9856     {\the\glslabeltok}%
9857 }

```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```

9858 \newabbreviationstyle{short-postlong-user-desc}%
9859 {%
9860   \renewcommand*{\CustomAbbreviationFields}{%
9861     name={\glxtrshortlonguserdescname},
9862     sort={\the\glssshorttok},
9863     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9864     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9865     text={\protect\glssabbrvuserfont{\the\glssshorttok}},%
9866     plural={\protect\glssabbrvuserfont{\the\glssshortpltok}}}%
9867 }%
9868 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9869   \csdef{glxtrpostlink\glscategorylabel}{%
9870     \glxtrifwasfirstuse
9871     {%
9872       \glxtruserparen
9873         {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9874         {\glslabel}%
9875     }%
9876     }%
9877 }%
9878 \glshasattribute{\the\glslabeltok}{regular}%
9879 }%

```

```

9880     \glsetattribute{\the\glslabeltok}{regular}{false}%
9881   }%
9882   {}%
9883 }%
9884}%
9885{%
9886  \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
9887 }

```

short-user-desc

```

9888 \newabbreviationstyle{long-short-user-desc}%
9889 {%
9890   \renewcommand*{\CustomAbbreviationFields}{%
9891     name={\glxtrlongshortuserdescname},
9892     sort={\glxtrlongshortdescsort},%

9893     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9894       \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9895     {\the\glslabeltok}},%
9896     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9897       \protect\glsxtruserparen
9898       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9899     text={\protect\glsabbrvfont{\the\glsshorttok}},%
9900     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9901   }%

```

Unset the regular attribute if it has been set.

```

9902   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9903     \glshasattribute{\the\glslabeltok}{regular}%
9904     {%
9905       \glsetattribute{\the\glslabeltok}{regular}{false}%
9906     }%
9907     {}%
9908   }%
9909}%
9910{%
9911  \GlsXtrUseAbbrStyleFmts{long-short-user}%
9912 }

```

short-long-user

```

9913 \newabbreviationstyle{short-long-user}%
9914 {%

  \glslonguserfont is used in the description since \glsdesc doesn't set the style. (Now in
  \glsuserdescription.)

9915   \renewcommand*{\CustomAbbreviationFields}{%
9916     name={\glxtrshortlongname},
9917     sort={\the\glsshorttok},
9918     description={\protect\glsuserdescription{\the\glslongtok}%
9919       {\the\glslabeltok}},%

```



```

9920 first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
9921 \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9922 {\the\glslabeltok}},%
9923 firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
9924 \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9925 {\the\glslabeltok}},%

```

```

9926 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

9927 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9928 \glshasattribute{\the\glslabeltok}{regular}%
9929 {%
9930 \glssetattribute{\the\glslabeltok}{regular}{false}%
9931 }%
9932 {}%
9933 }%
9934 }%
9935 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9936 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9937 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9938 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9939 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9940 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9941 \renewcommand*{\glsxtrfullformat}[2]{%
9942 \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
9943 \ifglsxtrinsertinside\else##2\fi
9944 \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
9945 }%
9946 \renewcommand*{\glsxtrfullplformat}[2]{%
9947 \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
9948 \ifglsxtrinsertinside\else##2\fi
9949 \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
9950 }%
9951 \renewcommand*{\Glsxtrfullformat}[2]{%
9952 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
9953 \ifglsxtrinsertinside\else##2\fi
9954 \glsxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9955 }%
9956 \renewcommand*{\Glsxtrfullplformat}[2]{%
9957 \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
9958 \ifglsxtrinsertinside\else##2\fi
9959 \glsxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
9960 }%
9961 }

```

-long-user-desc

```
9962 \newabbreviationstyle{short-long-user-desc}%
9963 {%
9964   \renewcommand*{\CustomAbbreviationFields}{%
9965     name={\glxtrshortlonguserdescname},
9966     sort={\glxtrshortlongdescsort},%

9967     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
9968       \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9969     {\the\glslabeltok}},%
9970     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
9971       \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9972     {\the\glslabeltok}},%
9973     text={\protect\glssabrvfont{\the\glsshorttok}},%
9974     plural={\protect\glssabrvfont{\the\glsshortpltok}}}%
9975   }%

   Unset the regular attribute if it has been set.
9976   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9977     \glshasattribute{\the\glslabeltok}{regular}%
9978     {%
9979       \glissetattribute{\the\glslabeltok}{regular}{false}%
9980     }%
9981   }%
9982 }%
9983 }%
9984 {%
9985   \GlsXtrUseAbbrStyleFmts{short-long-user}%
9986 }
```

1.7.7 Predefined Styles (Hyphen)

These styles are designed to work with the `markwords` attribute. They check if the inserted material (provided by the final optional argument of commands like `\gls`) starts with a hyphen. If it does, the insert is added to the parenthetical material. Note that commands like `\glxtrlong` set `\glinsert` to empty with the entire link-text stored in `\glscustomtext`.

`trifhyphenstart` Checks if the argument starts with a hyphen. The argument may be `\glinsert` so check for that and expand.

```
9987 \newrobustcmd*{\glxtrifhyphenstart}[3]{%
9988   \ifx\glinsert#1\relax
9989     \expandafter\@glxtrifhyphenstart#1\relax\relax
9990     \@end@glxtrifhyphenstart{#2}{#3}%
9991   \else
9992     \@glxtrifhyphenstart#1\relax\relax\@end@glxtrifhyphenstart{#2}{#3}%
9993   \fi
9994 }
```

trifhyphenstart

```
9995 \def\@glxtrifhyphenstart#1#2\@end@glxtrifhyphenstart#3#4{%
9996   \ifx-#1\relax#3\else #4\fi
9997 }
```

rlonghyphenshort

```
\glxtrlonghyphenshort{<label>}{<long>}{<short>}{<insert>}
```

The *<long>* and *<short>* arguments may be the plural form. The *<long>* argument may also be the first letter uppercase form.

```
9998 \newcommand*\glxtrlonghyphenshort}[4]{%
```

Grouping is needed to localise the redefinitions.

```
9999  {%
```

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.

```
10000   \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{ }%
10001   \glsfirslonghyphenfont{#2\ifglxtrininsertinside{#4}\fi}%
10002   \ifglxtrininsertinside\else{#4}\fi
10003   \glxtrfullsep{#1}%
10004   \glxtrparen{\glsfirstabbrvhyphenfont{#3\ifglxtrininsertinside{#4}\fi}%
10005     \ifglxtrininsertinside\else{#4}\fi}%
10006 }%
10007 }
```

abbrvhyphenfont

```
10008 \newcommand*\glsabbrvhyphenfont{\glsabbrvdefaultfont}%
```

abbrvhyphenfont

```
10009 \newcommand*\glsfirstabbrvhyphenfont{\glsabbrvhyphenfont}%
```

slonghyphenfont

```
10010 \newcommand*\glslonghyphenfont{\glslongdefaultfont}%
```

tlonghyphenfont

```
10011 \newcommand*\glsfirslonghyphenfont{\glslonghyphenfont}%
```

The default short form suffix:

xtrhyphensuffix

```
10012 \newcommand*\glxtrhyphensuffix{\glxtrabbrvpluralsuffix}
```

en-short-hyphen Designed for use with the `markwords` attribute.

```
10013 \newabbreviationstyle{long-hyphen-short-hyphen}%
10014 {%
```

```

10015 \renewcommand*{\CustomAbbreviationFields}{%
10016   name={\glxtrlongshortname},
10017   sort={\the\glsshorttok},
10018   first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
10019     \protect\glxtrfullsep{\the\glslabeltok}%
10020     \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
10021   firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
10022     \protect\glxtrfullsep{\the\glslabeltok}%
10023     \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
10024   plural={\protect\glssabbrvhyphenfont{\the\glsshortpltok}}},%
10025   description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10026 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10027   \glshasattribute{\the\glslabeltok}{regular}%
10028   {%
10029     \glissetattribute{\the\glslabeltok}{regular}{false}%
10030   }%
10031   {}%
10032 }%
10033}%
10034{%
10035 \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
10036 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvhyphenfont{##1}}%
10037 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10038 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10039 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10040 \renewcommand*{\glxtrfullformat}[2]{%
10041   \glxtrlonghyphenshort{##1}{\glssaccesslong{##1}}{\glssaccessshort{##1}}{##2}%
10042 }%
10043 \renewcommand*{\glxtrfullplformat}[2]{%
10044   \glxtrlonghyphenshort{##1}{\glssaccesslongpl{##1}}%
10045   {\glssaccessshortpl{##1}}{##2}%
10046 }%
10047 \renewcommand*{\Glsxtrfullformat}[2]{%
10048   \glxtrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\Glsaccessshort{##1}}{##2}%
10049 }%
10050 \renewcommand*{\Glsxtrfullplformat}[2]{%
10051   \glxtrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
10052   {\Glsaccessshortpl{##1}}{##2}%
10053 }%
10054}

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

10055 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
10056 {%
10057   \renewcommand*{\CustomAbbreviationFields}{%
10058     name={\glxtrlongshortdescname},

```

```

10059 sort={\glxtrlongshortdescsort},
10060 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
10061 \protect\glxtrfullsep{\the\glslabeltok}%
10062 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
10063 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
10064 \protect\glxtrfullsep{\the\glslabeltok}%
10065 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
10066 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10067 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
10068 }%

```

Unset the regular attribute if it has been set.

```

10069 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10070 \glsasattribute{\the\glslabeltok}{regular}%
10071 {%
10072 \glssetattribute{\the\glslabeltok}{regular}{false}%
10073 }%
10074 {}%
10075 }%
10076 }%
10077 {%
10078 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
10079 }

```

onghyphennoshort `\glxtrlonghyphennoshort{<label>}{<long>}{<insert>}`

```

10080 \newcommand*{\glxtrlonghyphennoshort}[3]{%

```

Grouping is needed to localise the redefinitions.

```

10081 {%

```

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.

```

10082 \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{%
10083 \glsfirstlonghyphenfont{#2\ifglxtrininsertinside{#3}\fi}%
10084 \ifglxtrininsertinside\else{#3}\fi
10085 }%
10086 }

```

hort-desc-noreg This version doesn't show the short form (except explicitly with `\glxtrshort`). Since `\glxtrshort` doesn't support the hyphen switch, the short form just uses the default short-form font command. This style won't work with the regular as the regular form isn't flexible enough.

```

10087 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
10088 {%
10089 \renewcommand*{\CustomAbbreviationFields}{%

```

```

10090     name={\glxtrlongnosshortdescname},
10091     sort={\expandonce\glxtrorglong},
10092     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10093     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10094     plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
10095 }%

```

Unset the regular attribute if it has been set.

```

10096 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10097   \glshasattribute{\the\glslabeltok}{regular}%
10098   {%
10099     \glissetattribute{\the\glslabeltok}{regular}{false}%
10100   }%
10101   {}}%
10102 }%
10103 }%
10104 {%
10105   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10106 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10107 \renewcommand*{\glssabrvfont[1]}{\glssabrvdefaultfont{##1}}%
10108 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
10109 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10110 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10111 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10112   \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
10113 }%
10114 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10115   \glxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
10116 }%
10117 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10118   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
10119 }%
10120 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10121   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
10122 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10123 \renewcommand*{\glxtrinlinefullformat}[2]{%
10124   \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
10125   \glxtrfullsep{##1}%
10126   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshort{##1}}}%
10127 }%
10128 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10129   \glxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
10130   \glxtrfullsep{##1}%
10131   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{##1}}}%
10132 }%

```

```

10133 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10134   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
10135   \glsxtrfullsep{##1}%
10136   \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
10137 }%
10138 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10139   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
10140   \glsxtrfullsep{##1}%
10141   \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
10142 }%

```

The first use full form only displays the long form.

```

10143 \renewcommand*{\glsxtrfullformat}[2]{%
10144   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
10145 }%
10146 \renewcommand*{\glsxtrfullplformat}[2]{%
10147   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
10148 }%
10149 \renewcommand*{\Glsxtrfullformat}[2]{%
10150   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
10151 }%
10152 \renewcommand*{\Glsxtrfullplformat}[2]{%
10153   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
10154 }%
10155 }

```

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.

```

10156 \newabbreviationstyle{long-hyphen-noshort-noreg}%
10157 {%
10158   \renewcommand*{\CustomAbbreviationFields}{%
10159     name={\glsxtrlongnoshortname},
10160     sort={\the\glsshorttok},
10161     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10162     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10163     text={\protect\glslonghyphenfont{\the\glslongtok}},%
10164     plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
10165     description={\the\glslongtok}%
10166   }%

```

Unset the regular attribute if it has been set.

```

10167 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10168   \glshasattribute{\the\glslabeltok}{regular}%
10169   {%
10170     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
10171   }%
10172   {}%
10173 }%
10174 }%

```

```

10175 {%
10176   \GlsXtrUseAbbrStyleFmts{long-desc}%
10177 }

```

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.

```

10178 \newcommand*{\glsxtrlonghyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

10179  {%
10180    \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
10181    \glsfirstlonghyphenfont{#1}%
10182  }%
10183 }

```

rpostthyphenshort `\glsxtrpostthyphenshort{<label>}{<insert>}`

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like `\glsxtrlonghyphenshort` but omits the *<long>* part. This always uses the singular short form.

```

10184 \newcommand*{\glsxtrpostthyphenshort}[2]{%
10185  {%
10186    \glsxtrifhyphenstart{#2}{\def\glsxtrwordsep{-}}{}%
10187    \ifglsxtrininsertinside{\glsfirstlonghyphenfont{#2}}\else{#2}\fi
10188    \glsxtrfullsep{#1}%
10189    \glsxtrparen
10190    {\glsfirstabbrvhyphenfont{\glsentryshort{#1}}\ifglsxtrininsertinside{#2}\fi}%
10191    \ifglsxtrininsertinside\else{#2}\fi
10192  }%
10193 }%
10194 }

```

hyphensubsequent `\glsxtrpostthyphensubsequent{<label>}{<insert>}`

Format in the post-link hook for subsequent use. The label is ignored by default.

```

10195 \newcommand*{\glsxtrpostthyphensubsequent}[2]{%
10196   \glsabbrvfont{\ifglsxtrininsertinside {#2}\fi}%
10197   \ifglsxtrininsertinside \else{#2}\fi
10198 }

```


ostshort-hyphen Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

10199 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
10200 {%
10201   \renewcommand*{\CustomAbbreviationFields}{%
10202     name={\glxtrlongshortname},
10203     sort={\the\glsshorttok},
10204     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10205     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10206     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
10207     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10208   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10209     \csdef{glsxtrpostlink\glscategorylabel}{%
10210       \glxtrifwasfirstuse
10211       {%
10212         \glxtrposthyphenshort{\glslabel}{\glsinsert}%
10213       }%
10214     }%

```

Put the insertion into the post-link:

```

10215       \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10216     }%
10217   }%
10218   \glshasattribute{\the\glslabeltok}{regular}%
10219   {%
10220     \glissetattribute{\the\glslabeltok}{regular}{false}%
10221   }%
10222   {}%
10223 }%
10224 }%
10225 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10226 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10227 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10228 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10229 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10230 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10231 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10232   \glsabbrvfont{\glsaccessshort{##1}}%
10233 }%
10234 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10235   \glsabbrvfont{\glsaccessshortpl{##1}}%
10236 }%
10237 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10238   \glsabbrvfont{\Glsaccessshort{##1}}%
10239 }%
10240 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%

```

```

10241 \glsabbrvfont{\Glsaccessshortpl{##1}}%
10242 }%
    First use full form:
10243 \renewcommand*{\glsxtrfullformat}[2]{%
10244 \glsxtrlonghyphen{\glsaccesslong{##1}}{##1}{##2}%
10245 }%
10246 \renewcommand*{\glsxtrfullplformat}[2]{%
10247 \glsxtrlonghyphen{\glsaccesslongpl{##1}}{##1}{##2}%
10248 }%
10249 \renewcommand*{\Glsxtrfullformat}[2]{%
10250 \glsxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
10251 }%
10252 \renewcommand*{\Glsxtrfullplformat}[2]{%
10253 \glsxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
10254 }%

```

In-line format.

```

10255 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10256 \glsfirstlonghyphenfont{\glsaccesslong{##1}}%
10257 \ifglsxtrininsertinside{##2}\fi}%
10258 \ifglsxtrininsertinside \else{##2}\fi
10259 }%
10260 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10261 \glsfirstlonghyphenfont{\glsaccesslongpl{##1}}%
10262 \ifglsxtrininsertinside{##2}\fi}%
10263 \ifglsxtrininsertinside \else{##2}\fi
10264 }%
10265 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10266 \glsfirstlonghyphenfont{\Glsaccesslong{##1}}%
10267 \ifglsxtrininsertinside{##2}\fi}%
10268 \ifglsxtrininsertinside \else{##2}\fi
10269 }%
10270 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10271 \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}}%
10272 \ifglsxtrininsertinside{##2}\fi}%
10273 \ifglsxtrininsertinside \else{##2}\fi
10274 }%
10275 }

```

`ort-hyphen-desc` Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

10276 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
10277 {%
10278 \renewcommand*{\CustomAbbreviationFields}{%
10279 name={\glsxtrlongshortdescname},
10280 sort={\glsxtrlongshortdescsort},%
10281 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10282 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10283 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10284 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%

```

```

10285 }%
10286 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10287   \csdef{glxtrpostlink\glscategorylabel}{%
10288     \glxtrifwasfirstuse
10289     {%
10290       \glxtrposthyphenshort{\glslabel}{\glsinsert}}%
10291     }%
10292     {%

```

Put the insertion into the post-link:

```

10293       \glxtrposthyphensubsequent{\glslabel}{\glsinsert}}%
10294     }%
10295   }%
10296   \glshasattribute{\the\glslabeltok}{regular}%
10297   {%
10298     \glssetattribute{\the\glslabeltok}{regular}{false}%
10299   }%
10300   {}%
10301 }%
10302 }%
10303 {%
10304   \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
10305 }

```

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.

```
10306 \newcommand*{\glxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
10307 {%
```

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.)

```

10308   \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{%
10309   \glsfirstabbrvhyphenfont{#2\ifglxtrininsertinside{#4}\fi}%
10310   \ifglxtrininsertinside\else{#4}\fi
10311   \glxtrfullsep{#1}%
10312   \glxtrparen{\glsfirstlonghyphenfont{#3\ifglxtrininsertinside{#4}\fi}%
10313   \ifglxtrininsertinside\else{#4}\fi}%
10314 }%
10315 }

```

hen-long-hyphen Designed for use with the markwords attribute.

```
10316 \newabbreviationstyle{short-hyphen-long-hyphen}%
```

```

10317 {%
10318   \renewcommand*{\CustomAbbreviationFields}{%
10319     name={\glxtrshortlongname},
10320     sort={\the\glsshorttok},
10321     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
10322       \protect\glxtrfullsep{\the\glslabeltok}%
10323       \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10324     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
10325       \protect\glxtrfullsep{\the\glslabeltok}%
10326       \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10327     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}},%
10328     description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10329   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10330     \glshasattribute{\the\glslabeltok}{regular}%
10331     {%
10332       \glissetattribute{\the\glslabeltok}{regular}{false}%
10333     }%
10334   }%
10335 }%
10336 }%
10337 {%
10338   \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
10339   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10340   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10341   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10342   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10343   \renewcommand*{\glxtrfullformat}[2]{%
10344     \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
10345   }%
10346   \renewcommand*{\glxtrfullplformat}[2]{%
10347     \glxtrshorthyphenlong{##1}%
10348     {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
10349   }%
10350   \renewcommand*{\Glsxtrfullformat}[2]{%
10351     \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
10352   }%
10353   \renewcommand*{\Glsxtrfullplformat}[2]{%
10354     \glxtrshorthyphenlong{##1}%
10355     {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
10356   }%
10357 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

10358 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
10359 {%
10360   \renewcommand*{\CustomAbbreviationFields}{%

```

```

10361   name={\glxtrshortlongdescname},
10362   sort={\glxtrshortlongdescsort},
10363   first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
10364     \protect\glxtrfullsep{\the\glslabeltok}%
10365     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10366   firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
10367     \protect\glxtrfullsep{\the\glslabeltok}%
10368     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10369   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10370   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
10371 }%

```

Unset the regular attribute if it has been set.

```

10372 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10373   \glshasattribute{\the\glslabeltok}{regular}%
10374   {%
10375     \glissetattribute{\the\glslabeltok}{regular}{false}%
10376   }%
10377   {}%
10378 }%
10379 }%
10380 {%
10381   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
10382 }

```

`\glxtrshorthyphen` `\glxtrshorthyphen{<short>}{<label>}{<insert>}`

Used by short-hyphen-postlong-hyphen. The `<insert>` is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```

10383 \newcommand*{\glxtrshorthyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

10384 {%
10385   \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}}%
10386   \glsfirstabbrvhyphenfont{#1}%
10387 }%
10388 }

```

`\glxtrposthyphenlong` `\glxtrposthyphenlong{<label>}{<insert>}`

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glxtrshorthyphenlong` but omits the `<short>` part. This always uses the singular long form.

```

10389 \newcommand*{\glxtrposthyphenlong}[2]{%
10390   {%

```

```

10391 \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
10392 \ifglxtrininsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
10393 \glxtrfullsep{#1}%
10394 \glxtrparen
10395 {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglxtrininsertinside{#2}\fi}%
10396 \ifglxtrininsertinside\else{#2}\fi
10397 }%
10398 }%
10399 }

```

postlong-hyphen Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

10400 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
10401 {%
10402 \renewcommand*{\CustomAbbreviationFields}{%
10403 name={\glxtrshortlongname},
10404 sort={\the\glsshorttok},
10405 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10406 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10407 plural={\protect\glssabbrvhyphenfont{\the\glsshortpltok}},%
10408 description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10409 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10410 \csdef{glxtrpostlink\glscategorylabel}{%
10411 \glxtrifwasfirstuse
10412 {%
10413 \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10414 }%
10415 }%

```

Put the insertion into the post-link:

```

10416 \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10417 }%
10418 }%
10419 \glshasattribute{\the\glslabeltok}{regular}%
10420 {%
10421 \glissetattribute{\the\glslabeltok}{regular}{false}%
10422 }%
10423 {}%
10424 }%
10425 }%
10426 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10427 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10428 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvhyphenfont{##1}}%
10429 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10430 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10431 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10432 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10433   \glsabbrvfont{\glsaccessshort{##1}}}%
10434 }%
10435 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10436   \glsabbrvfont{\glsaccessshortpl{##1}}}%
10437 }%
10438 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10439   \glsabbrvfont{\Glsaccessshort{##1}}}%
10440 }%
10441 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10442   \glsabbrvfont{\Glsaccessshortpl{##1}}}%
10443 }%

```

First use full form:

```

10444 \renewcommand*{\glxtrfullformat}[2]{%
10445   \glxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
10446 }%
10447 \renewcommand*{\glxtrfullplformat}[2]{%
10448   \glxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
10449 }%
10450 \renewcommand*{\Glsxtrfullformat}[2]{%
10451   \glxtrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
10452 }%
10453 \renewcommand*{\Glsxtrfullplformat}[2]{%
10454   \glxtrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
10455 }%

```

In-line format. Commands like `\glxtrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

10456 \renewcommand*{\glxtrinlinefullformat}[2]{%
10457   \glsfirstabbrvhyphenfont{\glsaccessshort{##1}}%
10458   \ifglxtrininsertinside{##2}\fi}%
10459   \ifglxtrininsertinside \else{##2}\fi
10460 }%
10461 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10462   \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}}%
10463   \ifglxtrininsertinside{##2}\fi}%
10464   \ifglxtrininsertinside \else{##2}\fi
10465 }%
10466 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10467   \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}}%
10468   \ifglxtrininsertinside{##2}\fi}%
10469   \ifglxtrininsertinside \else{##2}\fi
10470 }%
10471 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10472   \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}}%
10473   \ifglxtrininsertinside{##2}\fi}%
10474   \ifglxtrininsertinside \else{##2}\fi
10475 }%
10476 }

```

ong-hyphen-desc Like short-hyphen-postlong-hyphen but the description must be supplied by the user.

```
10477 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
10478 {%
10479   \renewcommand*{\CustomAbbreviationFields}{%
10480     name={\glsxtrshortlongdescname},
10481     sort={\glsxtrshortlongdescsort},%
10482     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10483     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10484     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10485     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10486   }%
10487   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10488     \csdef{glsxtrpostlink\glscategorylabel}{%
10489       \glsxtrifwasfirstuse
10490       {%
10491         \glsxtrposthyphenlong{\glslabel}{\glsinsert}%
10492       }%
10493     }%
```

Put the insertion into the post-link:

```
10494       \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10495     }%
10496   }%
10497   \glsattribute{\the\glslabeltok}{regular}%
10498   {%
10499     \glssetattribute{\the\glslabeltok}{regular}{false}%
10500   }%
10501   {}%
10502 }%
10503 }%
10504 {%
10505   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
10506 }
```

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

```
10507 \newcommand*{\glsabbrvonlyfont}{\glsabbrvdefaultfont}%
```

stabbrvonlyfont

```
10508 \newcommand*{\glsfirstabbrvonlyfont}{\glsabbrvonlyfont}%
```

glslongonlyfont

```
10509 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%
```

rstlongonlyfont

```
10510 \newcommand*{\glsfirstlongonlyfont}{\glslongonlyfont}%
```


The default short form suffix:

lsxtronlysuffix

```
10511 \newcommand*{\glxtronlysuffix}{\glxtrabbrvpluralsuffix}
```

\glxtronlyname The default name format for this style.

```
10512 \newcommand*{\glxtronlyname}{%
10513   \protect\glxtronlyfont{\the\glsshorttok}%
10514 }
```

only-short-only

```
10515 \newabbreviationstyle{long-only-short-only}%
10516 {%
10517   \renewcommand*{\CustomAbbreviationFields}{%
10518     name={\glxtronlyname},
10519     sort={\the\glsshorttok},
10520     first={\protect\glxtronlyfont{\the\glslongtok}},%
10521     firstplural={\protect\glxtronlyfont{\the\glslongpltok}},%
10522     plural={\protect\glxtronlyfont{\the\glsshortpltok}},%
10523     description={\protect\glxtronlyfont{\the\glslongtok}}}%
10524 }
```

Unset the regular attribute if it has been set.

```
10524 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10525   \glshasattribute{\the\glslabeltok}{regular}%
10526   {%
10527     \glissetattribute{\the\glslabeltok}{regular}{false}%
10528   }%
10529   {}%
10530 }%
10531 }%
10532 {%
10533   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtronlysuffix}%
10534   \renewcommand*{\glxtronlyfont}[1]{\glxtronlyfont{##1}}%
10535   \renewcommand*{\glxtronlyfont}[1]{\glxtronlyfont{##1}}%
10536   \renewcommand*{\glxtronlyfont}[1]{\glxtronlyfont{##1}}%
10537   \renewcommand*{\glxtronlyfont}[1]{\glxtronlyfont{##1}}%
10538 }
```

The first use full form doesn't show the short form.

```
10538 \renewcommand*{\glxtrfullformat}[2]{%
10539   \glxtronlyfont{\glxtronlyfont{##1}}\ifglxtrinsertinside##2\fi}%
10540   \ifglxtrinsertinside\else##2\fi
10541 }%
10542 \renewcommand*{\glxtrfullplformat}[2]{%
10543   \glxtronlyfont{\glxtronlyfont{##1}}\ifglxtrinsertinside##2\fi}%
10544   \ifglxtrinsertinside\else##2\fi
10545 }%
10546 \renewcommand*{\GlsXtrfullformat}[2]{%
10547   \glxtronlyfont{\glxtronlyfont{##1}}\ifglxtrinsertinside##2\fi}%
10548   \ifglxtrinsertinside\else##2\fi
10549 }%
```

```

10550 \renewcommand*{\Glsxtrfullplformat}[2]{%
10551   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10552   \ifglxtrinsertinside\else##2\fi
10553 }%

```

The inline full form does show the short form.

```

10554 \renewcommand*{\glxtrinlinefullformat}[2]{%
10555   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10556   \ifglxtrinsertinside\else##2\fi
10557   \glxtrfullsep{##1}%
10558   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshort{##1}}}%
10559 }%
10560 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10561   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10562   \ifglxtrinsertinside\else##2\fi
10563   \glxtrfullsep{##1}%
10564   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10565 }%
10566 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10567   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10568   \ifglxtrinsertinside\else##2\fi
10569   \glxtrfullsep{##1}%
10570   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10571 }%
10572 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10573   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10574   \ifglxtrinsertinside\else##2\fi
10575   \glxtrfullsep{##1}%
10576   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10577 }%
10578 }

```

xtronlydescsort

```

10579 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

10580 \newcommand*{\glxtronlydescname}{%
10581   \protect\glslongfont{\the\glslongtok}}%
10582 }

```

short-only-desc

```

10583 \newabbreviationstyle{long-only-short-only-desc}%
10584 {%
10585   \renewcommand*{\CustomAbbreviationFields}{%
10586     name={\glxtronlydescname},
10587     sort={\glxtronlydescsort},%
10588     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10589     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10590     text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%

```

```

10591 plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}}%
10592 }%

Unset the regular attribute if it has been set.

10593 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10594   \glshasattribute{\the\glslabeltok}{regular}%
10595   {%
10596     \glssetattribute{\the\glslabeltok}{regular}{false}%
10597   }%
10598   {}}%
10599 }%
10600}%
10601{%
10602   \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
10603 }

```

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:

```
10604 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```

10605 \renewcommand*{\markright}[1]{%
10606   \glsxtrmarkhook

```

```

10607 \@glsxtr@org@markright{\@glsxtrinmark#1\@glsxtrnotinmark}%
10608 \glsxtrrestoremarkhook
10609 }

```

\markboth Save original definition:

```

10610 \let\@glsxtr@org@markboth\markboth

```

Redefine (grouping not added in case it interferes with the original code):

```

10611 \renewcommand*{\markboth}[2]{%
10612 \glsxtrmarkhook
10613 \@glsxtr@org@markboth
10614 {\@glsxtrinmark#1\@glsxtrnotinmark}%
10615 {\@glsxtrinmark#2\@glsxtrnotinmark}%
10616 \glsxtrrestoremarkhook
10617 }

```

Also do this for \@starttoc

\@starttoc Save original definition:

```

10618 \let\@glsxtr@org@@starttoc\@starttoc

```

Redefine:

```

10619 \renewcommand*{\@starttoc}[1]{%
10620 \glsxtrmarkhook
10621 \@glsxtrinmark
10622 \@glsxtr@org@@starttoc{#1}%
10623 \@glsxtrnotinmark
10624 \glsxtrrestoremarkhook
10625 }

```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```

10626 \newcommand*{\glsxtrRevertMarks}{%
10627 \let\markright\@glsxtr@org@markright
10628 \let\markboth\@glsxtr@org@markboth
10629 \let\@starttoc\@glsxtr@org@@starttoc
10630 }

```

\glsxtrifinmark

```

10631 \newcommand*{\glsxtrifinmark}[2]{#2}

```

\@glsxtrinmark

```

10632 \newrobustcmd*{\@glsxtrinmark}{%
10633 \let\glsxtrifinmark\@firstoftwo
10634 }

```

glsxtrnotinmark

```

10635 \newrobustcmd*{\@glsxtrnotinmark}{%
10636 \let\glsxtrifinmark\@secondoftwo
10637 }

```

eorpdforheading

```
10638 \ifdef\texorpdfstring
10639 {
10640   \newcommand*\glxstrtitleorpdforheading[3]{\texorpdfstring{#1}{#2}}
10641 }
10642 {
10643   \newcommand*\glxstrtitleorpdforheading[3]{#1}
10644 }
```

\glxstrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```
10645 \newcommand*\glxstrmarkhook}{%
```

Save current definitions:

```
10646 \let\@glxstr@org@MakeUppercase\MakeUppercase
10647 \let\@glxstr@org@glxstrtitleorpdforheading\glxstrtitleorpdforheading
10648 \let\@glxstr@org@glxstrtitleshort\glxstrtitleshort
10649 \let\@glxstr@org@glxstrtitleshortpl\glxstrtitleshortpl
10650 \let\@glxstr@org@Glsxstrtitleshort\Glsxstrtitleshort
10651 \let\@glxstr@org@Glsxstrtitleshortpl\Glsxstrtitleshortpl
10652 \let\@glxstr@org@glxstrtitlename\glxstrtitlename
10653 \let\@glxstr@org@Glsxstrtitlename\Glsxstrtitlename
10654 \let\@glxstr@org@glxstrtitletext\glxstrtitletext
10655 \let\@glxstr@org@Glsxstrtitletext\Glsxstrtitletext
10656 \let\@glxstr@org@glxstrtitleplural\glxstrtitleplural
10657 \let\@glxstr@org@Glsxstrtitleplural\Glsxstrtitleplural
10658 \let\@glxstr@org@glxstrtitlefirst\glxstrtitlefirst
10659 \let\@glxstr@org@Glsxstrtitlefirst\Glsxstrtitlefirst
10660 \let\@glxstr@org@glxstrtitlefirstplural\glxstrtitlefirstplural
10661 \let\@glxstr@org@Glsxstrtitlefirstplural\Glsxstrtitlefirstplural
10662 \let\@glxstr@org@glxstrtitlelong\glxstrtitlelong
10663 \let\@glxstr@org@glxstrtitlelongpl\glxstrtitlelongpl
10664 \let\@glxstr@org@Glsxstrtitlelong\Glsxstrtitlelong
10665 \let\@glxstr@org@Glsxstrtitlelongpl\Glsxstrtitlelongpl
10666 \let\@glxstr@org@glxstrtitlefull\glxstrtitlefull
10667 \let\@glxstr@org@glxstrtitlefullpl\glxstrtitlefullpl
10668 \let\@glxstr@org@Glsxstrtitlefull\Glsxstrtitlefull
10669 \let\@glxstr@org@Glsxstrtitlefullpl\Glsxstrtitlefullpl
```

New definitions

```
10670 \let\glxstrifinmark\@firstoftwo
10671 \let\MakeUppercase\MakeTextUppercase
10672 \let\glxstrtitleorpdforheading\@thirdofthree
10673 \let\glxstrtitleshort\glxstrheadshort
10674 \let\glxstrtitleshortpl\glxstrheadshortpl
10675 \let\Glsxstrtitleshort\Glsxstrheadshort
10676 \let\Glsxstrtitleshortpl\Glsxstrheadshortpl
10677 \let\glxstrtitlename\glxstrheadname
10678 \let\Glsxstrtitlename\Glsxstrheadname
10679 \let\glxstrtitletext\glxstrheadtext
```

```

10680 \let\Glsxtrtitletext\Glsxtrheadtext
10681 \let\glxstrtitleplural\glxstrheadplural
10682 \let\Glsxtrtitleplural\Glsxtrheadplural
10683 \let\glxstrtitlefirst\glxstrheadfirst
10684 \let\Glsxtrtitlefirst\Glsxtrheadfirst
10685 \let\glxstrtitlefirstplural\glxstrheadfirstplural
10686 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
10687 \let\glxstrtitlelong\glxstrheadlong
10688 \let\glxstrtitlelongpl\glxstrheadlongpl
10689 \let\Glsxtrtitlelong\Glsxtrheadlong
10690 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
10691 \let\glxstrtitlefull\glxstrheadfull
10692 \let\glxstrtitlefullpl\glxstrheadfullpl
10693 \let\Glsxtrtitlefull\Glsxtrheadfull
10694 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
10695 }

```

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.)

```

10696 \newcommand*{\glxstrrestoremarkhook}{%
10697 \let\glxstrifinmark\@secondoftwo
10698 \let\MakeUppercase\@glxstr@org@MakeUppercase
10699 \let\glxstrtitleorpdforheading\@glxstr@org@glxstrtitleorpdforheading
10700 \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
10701 \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
10702 \let\Glsxtrtitleshort\@glxstr@org@Glsxtrtitleshort
10703 \let\Glsxtrtitleshortpl\@glxstr@org@Glsxtrtitleshortpl
10704 \let\glxstrtitlename\@glxstr@org@glxstrtitlename
10705 \let\Glsxtrtitlename\@glxstr@org@Glsxtrtitlename
10706 \let\glxstrtitletext\@glxstr@org@glxstrtitletext
10707 \let\Glsxtrtitletext\@glxstr@org@Glsxtrtitletext
10708 \let\glxstrtitleplural\@glxstr@org@glxstrtitleplural
10709 \let\Glsxtrtitleplural\@glxstr@org@Glsxtrtitleplural
10710 \let\glxstrtitlefirst\@glxstr@org@glxstrtitlefirst
10711 \let\Glsxtrtitlefirst\@glxstr@org@Glsxtrtitlefirst
10712 \let\glxstrtitlefirstplural\@glxstr@org@glxstrtitlefirstplural
10713 \let\Glsxtrtitlefirstplural\@glxstr@org@Glsxtrtitlefirstplural
10714 \let\glxstrtitlelong\@glxstr@org@glxstrtitlelong
10715 \let\glxstrtitlelongpl\@glxstr@org@glxstrtitlelongpl
10716 \let\Glsxtrtitlelong\@glxstr@org@Glsxtrtitlelong
10717 \let\Glsxtrtitlelongpl\@glxstr@org@Glsxtrtitlelongpl
10718 \let\glxstrtitlefull\@glxstr@org@glxstrtitlefull
10719 \let\glxstrtitlefullpl\@glxstr@org@glxstrtitlefullpl
10720 \let\Glsxtrtitlefull\@glxstr@org@Glsxtrtitlefull
10721 \let\Glsxtrtitlefullpl\@glxstr@org@Glsxtrtitlefullpl
10722 }

```

Instead of using one document-wide conditional, use headuc attribute to determine whether or not to use the all upper case form.

glsxtrheadshort Command used to display short form in the page header.

```
10723 \newcommand*{\glsxtrheadshort}[1]{%
10724   \protect\NoCaseChange
10725   {%
10726     \glsifattribute{#1}{headuc}{true}%
10727     {%
10728       \GLSxtrshort[noindex,hyper=false]{#1}[]%
10729     }%
10730     {%
10731       \glsxtrshort[noindex,hyper=false]{#1}[]%
10732     }%
10733   }%
10734 }
```

ltxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```
10735 \newrobustcmd*{\glsxtrtitleshort}[1]{%
10736   \glsxtrshort[noindex,hyper=false]{#1}[]%
10737 }
```

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.

```
10738 \newcommand*{\glsxtrheadshortpl}[1]{%
10739   \protect\NoCaseChange
10740   {%
10741     \glsifattribute{#1}{headuc}{true}%
10742     {%
10743       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10744     }%
10745     {%
10746       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10747     }%
10748   }%
10749 }
```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```
10750 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
10751   \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10752 }
```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```
10753 \newcommand*{\Glsxtrheadshort}[1]{%
10754   \protect\NoCaseChange
10755   {%
```

```

10756 \glsifattribute{#1}{headuc}{true}%
10757 {%
10758 \GLSxtrshort[noindex,hyper=false]{#1}[]%
10759 }%
10760 {%
10761 \GLSxtrshort[noindex,hyper=false]{#1}[]%
10762 }%
10763 }%
10764 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10765 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
10766 \GLSxtrshort[noindex,hyper=false]{#1}[]%
10767 }

```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```

10768 \newcommand*{\Glsxtrheadshortpl}[1]{%
10769 \protect\NoCaseChange
10770 {%
10771 \glsifattribute{#1}{headuc}{true}%
10772 {%
10773 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10774 }%
10775 {%
10776 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10777 }%
10778 }%
10779 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10780 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
10781 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10782 }

```

\glsxtrheadname As above but for the name value.

```

10783 \newcommand*{\glsxtrheadname}[1]{%
10784 \protect\NoCaseChange
10785 {%
10786 \glsifattribute{#1}{headuc}{true}%
10787 {%
10788 \GLSname[noindex,hyper=false]{#1}[]%
10789 }%
10790 {%
10791 \glsname[noindex,hyper=false]{#1}[]%
10792 }%

```



```

10793 }%
10794 }

```

`glsxtrtitlename` Command to display name value in section title and table of contents.

```

10795 \newrobustcmd*{\glsxtrtitlename}[1]{%
10796   \glsname[noindex,hyper=false]{#1}[]%
10797 }

```

`\Glsxtrheadname` First letter converted to upper case

```

10798 \newcommand*{\Glsxtrheadname}[1]{%
10799   \protect\NoCaseChange
10800   {%
10801     \glsifattribute{#1}{headuc}{true}%
10802     {%
10803       \GLSname[noindex,hyper=false]{#1}[]%
10804     }%
10805     {%
10806       \Glsname[noindex,hyper=false]{#1}[]%
10807     }%
10808   }%
10809 }

```

`Glsxtrtitlename` Command to display name value in section title and table of contents with the first letter changed to upper case.

```

10810 %\changes{1.21}{2017-11-03}{new}
10811 \newrobustcmd*{\Glsxtrtitlename}[1]{%
10812   \Glsname[noindex,hyper=false]{#1}[]%
10813 }

```

`\glsxtrheadtext` As above but for the text value.

```

10814 \newcommand*{\glsxtrheadtext}[1]{%
10815   \protect\NoCaseChange
10816   {%
10817     \glsifattribute{#1}{headuc}{true}%
10818     {%
10819       \GLStext[noindex,hyper=false]{#1}[]%
10820     }%
10821     {%
10822       \glstext[noindex,hyper=false]{#1}[]%
10823     }%
10824   }%
10825 }

```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```

10826 \newrobustcmd*{\glsxtrtitletext}[1]{%
10827   \glstext[noindex,hyper=false]{#1}[]%
10828 }

```

`\Glsxtrheadtext` First letter converted to upper case

```
10829 \newcommand*{\Glsxtrheadtext}[1]{%
10830   \protect\NoCaseChange
10831   {%
10832     \glsifattribute{#1}{headuc}{true}%
10833     {%
10834       \GLStext[noindex,hyper=false]{#1}[]%
10835     }%
10836     {%
10837       \Glstext[noindex,hyper=false]{#1}[]%
10838     }%
10839   }%
10840 }
```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```
10841 \newrobustcmd*{\Glsxtrtitletext}[1]{%
10842   \Glstext[noindex,hyper=false]{#1}[]%
10843 }
```

`lsxtrheadplural` As above but for the plural value.

```
10844 \newcommand*{\glxtrheadplural}[1]{%
10845   \protect\NoCaseChange
10846   {%
10847     \glsifattribute{#1}{headuc}{true}%
10848     {%
10849       \GLSplural[noindex,hyper=false]{#1}[]%
10850     }%
10851     {%
10852       \glsplural[noindex,hyper=false]{#1}[]%
10853     }%
10854   }%
10855 }
```

`sxtrtitleplural` Command to display plural value in section title and table of contents.

```
10856 \newrobustcmd*{\glxtrtitleplural}[1]{%
10857   \glsplural[noindex,hyper=false]{#1}[]%
10858 }
```

`lsxtrheadplural` Convert first letter to upper case.

```
10859 \newcommand*{\Glsxtrheadplural}[1]{%
10860   \protect\NoCaseChange
10861   {%
10862     \glsifattribute{#1}{headuc}{true}%
10863     {%
10864       \GLSplural[noindex,hyper=false]{#1}[]%
10865     }%
10866     {%
```

```

10867 \Glsplural[noindex,hyper=false]{#1}[]%
10868 }%
10869 }%
10870 }

```

gsxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```

10871 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
10872 \Glsplural[noindex,hyper=false]{#1}[]%
10873 }

```

glsxtrheadfirst As above but for the first value.

```

10874 \newcommand*{\glsxtrheadfirst}[1]{%
10875 \protect\NoCaseChange
10876 {%
10877 \glsifattribute{#1}{headuc}{true}%
10878 {%
10879 \GLSfirst[noindex,hyper=false]{#1}[]%
10880 }%
10881 {%
10882 \glsfirst[noindex,hyper=false]{#1}[]%
10883 }%
10884 }%
10885 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```

10886 \newrobustcmd*{\lsxtrtitlefirst}[1]{%
10887 \glsfirst[noindex,hyper=false]{#1}[]%
10888 }

```

Glsxtrheadfirst First letter converted to upper case

```

10889 \newcommand*{\Glsxtrheadfirst}[1]{%
10890 \protect\NoCaseChange
10891 {%
10892 \glsifattribute{#1}{headuc}{true}%
10893 {%
10894 \GLSfirst[noindex,hyper=false]{#1}[]%
10895 }%
10896 {%
10897 \Glsfirst[noindex,hyper=false]{#1}[]%
10898 }%
10899 }%
10900 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```

10901 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
10902 \Glsfirst[noindex,hyper=false]{#1}[]%
10903 }

```

`headfirstplural` As above but for the firstplural value.

```
10904 \newcommand*{\glxtrheadfirstplural}[1]{%
10905   \protect\NoCaseChange
10906   {%
10907     \glsifattribute{#1}{headuc}{true}%
10908     {%
10909       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10910     }%
10911     {%
10912       \glsfirstplural[noindex,hyper=false]{#1}[]%
10913     }%
10914   }%
10915 }
```

`itlefirstplural` Command to display firstplural value in section title and table of contents.

```
10916 \newrobustcmd*{\glxtrtitlefirstplural}[1]{%
10917   \glsfirstplural[noindex,hyper=false]{#1}[]%
10918 }
```

`headfirstplural` First letter converted to upper case

```
10919 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10920   \protect\NoCaseChange
10921   {%
10922     \glsifattribute{#1}{headuc}{true}%
10923     {%
10924       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10925     }%
10926     {%
10927       \Glsfirstplural[noindex,hyper=false]{#1}[]%
10928     }%
10929   }%
10930 }
```

`itlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
10931 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
10932   \Glsfirstplural[noindex,hyper=false]{#1}[]%
10933 }
```

`\glxtrheadlong` Command used to display long form in the page header.

```
10934 \newcommand*{\glxtrheadlong}[1]{%
10935   \protect\NoCaseChange
10936   {%
10937     \glsifattribute{#1}{headuc}{true}%
10938     {%
10939       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10940     }%
10941     {%
```

```

10942     \glsxtrlong[noindex,hyper=false]{#1}[]%
10943 }%
10944 }%
10945 }

```

`\glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```

10946 \newrobustcmd*{\glsxtrtitlelong}[1]{%
10947   \glsxtrlong[noindex,hyper=false]{#1}[]%
10948 }

```

`\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.

```

10949 \newcommand*{\lsxtrheadlongpl}[1]{%
10950   \protect\NoCaseChange
10951   {%
10952     \glsifattribute{#1}{headuc}{true}%
10953     {%
10954       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10955     }%
10956     {%
10957       \glsxtrlongpl[noindex,hyper=false]{#1}[]%
10958     }%
10959   }%
10960 }

```

`\sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```

10961 \newrobustcmd*{\sxtrtitlelongpl}[1]{%
10962   \glsxtrlongpl[noindex,hyper=false]{#1}[]%
10963 }

```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

10964 \newcommand*{\Glsxtrheadlong}[1]{%
10965   \protect\NoCaseChange
10966   {%
10967     \glsifattribute{#1}{headuc}{true}%
10968     {%
10969       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10970     }%
10971     {%
10972       \Glsxtrlong[noindex,hyper=false]{#1}[]%
10973     }%
10974   }%
10975 }

```

`\Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10976 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
10977   \Glsxtrlong[noindex,hyper=false]{#1}[]%
10978 }

```

lsxtrheadlongpl Command used to display plural long form in the page header with the first letter converted to upper case.

```

10979 \newcommand*{\Glsxtrheadlongpl}[1]{%
10980   \protect\NoCaseChange
10981   {%
10982     \glsifattribute{#1}{headuc}{true}%
10983     {%
10984       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10985     }%
10986     {%
10987       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10988     }%
10989   }%
10990 }

```

sxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10991 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
10992   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10993 }

```

\glsxtrheadfull Command used to display full form in the page header.

```

10994 \newcommand*{\glsxtrheadfull}[1]{%
10995   \protect\NoCaseChange
10996   {%
10997     \glsifattribute{#1}{headuc}{true}%
10998     {%
10999       \GLSxtrfull[noindex,hyper=false]{#1}[]%
11000     }%
11001     {%
11002       \glsxtrfull[noindex,hyper=false]{#1}[]%
11003     }%
11004   }%
11005 }

```

glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents.

```

11006 \newrobustcmd*{\glsxtrtitlefull}[1]{%
11007   \glsxtrfull[noindex,hyper=false]{#1}[]%
11008 }

```

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.

```

11009 \newcommand*{\glsxtrheadfullpl}[1]{%

```

```

11010 \protect\NoCaseChange
11011 {%
11012   \glsifattribute{#1}{headuc}{true}%
11013   {%
11014     \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
11015   }%
11016   {%
11017     \glsxtrfullpl[noindex,hyper=false]{#1}[]%
11018   }%
11019 }%
11020 }

```

sxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents.

```

11021 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
11022   \glsxtrfullpl[noindex,hyper=false]{#1}[]%
11023 }

```

\Glsxtrheadfull Command used to display full form in the page header with the first letter converted to upper case.

```

11024 \newcommand*{\Glsxtrheadfull}[1]{%
11025   \protect\NoCaseChange
11026   {%
11027     \glsifattribute{#1}{headuc}{true}%
11028     {%
11029       \GLSxtrfull[noindex,hyper=false]{#1}[]%
11030     }%
11031     {%
11032       \Glsxtrfull[noindex,hyper=false]{#1}[]%
11033     }%
11034   }%
11035 }

```

Glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

11036 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
11037   \Glsxtrfull[noindex,hyper=false]{#1}[]%
11038 }

```

lsxtrheadfullpl Command used to display plural full form in the page header with the first letter converted to upper case.

```

11039 \newcommand*{\Glsxtrheadfullpl}[1]{%
11040   \protect\NoCaseChange
11041   {%
11042     \glsifattribute{#1}{headuc}{true}%
11043     {%
11044       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
11045     }%
11046     {%

```

```

11047 \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
11048 }%
11049 }%
11050 }

```

`\sxttitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

11051 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
11052 \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
11053 }

```

`\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.

```

11054 \ifdef\texorpdfstring
11055 {
11056 \newcommand*{\glsfmtshort}[1]{%
11057 \texorpdfstring
11058 {\Glsxtrtitleshort{#1}}%
11059 {\Glsentryshort{#1}}%
11060 }
11061 }
11062 {
11063 \newcommand*{\glsfmtshort}[1]{%
11064 \Glsxtrtitleshort{#1}}
11065 }

```

Similarly for the plural version.

`\glsfmtshortpl`

```

11066 \ifdef\texorpdfstring
11067 {
11068 \newcommand*{\glsfmtshortpl}[1]{%
11069 \texorpdfstring
11070 {\Glsxtrtitleshortpl{#1}}%
11071 {\Glsentryshortpl{#1}}%
11072 }
11073 }
11074 {
11075 \newcommand*{\glsfmtshortpl}[1]{%
11076 \Glsxtrtitleshortpl{#1}}
11077 }

```

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).

```

11078 \ifdef\texorpdfstring
11079 {
11080 \newcommand*{\Glsfmtshort}[1]{%

```



```

11081 \texorpdfstring
11082 {\Glsxtrtitleshort{#1}}}%
11083 {\glsentryshort{#1}}}%
11084 }
11085 }
11086 {
11087 \newcommand*{\Glsfmtshort}[1]{%
11088 \Glsxtrtitleshort{#1}}
11089 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

11090 \ifdef\texorpdfstring
11091 {
11092 \newcommand*{\Glsfmtshortpl}[1]{%
11093 \texorpdfstring
11094 {\Glsxtrtitleshortpl{#1}}}%
11095 {\glsentryshortpl{#1}}}%
11096 }
11097 }
11098 {
11099 \newcommand*{\Glsfmtshortpl}[1]{%
11100 \Glsxtrtitleshortpl{#1}}
11101 }

```

`\glsfmtname` As above but for the name value.

```

11102 \ifdef\texorpdfstring
11103 {
11104 \newcommand*{\glsfmtname}[1]{%
11105 \texorpdfstring
11106 {\glsxtrtitlename{#1}}}%
11107 {\glsentryname{#1}}}%
11108 }
11109 }
11110 {
11111 \newcommand*{\glsfmtname}[1]{%
11112 \glsxtrtitlename{#1}}
11113 }

```

`\Glsfmtname` First letter converted to upper case.

```

11114 \ifdef\texorpdfstring
11115 {
11116 \newcommand*{\Glsfmtname}[1]{%
11117 \texorpdfstring
11118 {\Glsxtrtitlename{#1}}}%
11119 {\glsentryname{#1}}}%
11120 }
11121 }
11122 {
11123 \newcommand*{\Glsfmtname}[1]{%

```

```

11124 \Glsxtrtitle{#1}}
11125 }

```

`\glsfmttext` As above but for the text value.

```

11126 \ifdef\texorpdfstring
11127 {
11128   \newcommand*\glsfmttext[1]{%
11129     \texorpdfstring
11130     {\glsxtrtitle{#1}}%
11131     {\glsentrytext{#1}}%
11132   }
11133 }
11134 {
11135   \newcommand*\glsfmttext[1]{%
11136     \glsxtrtitle{#1}}
11137 }

```

`\Glsfmttext` First letter converted to upper case.

```

11138 \ifdef\texorpdfstring
11139 {
11140   \newcommand*\Glsfmttext[1]{%
11141     \texorpdfstring
11142     {\Glsxtrtitle{#1}}%
11143     {\glsentrytext{#1}}%
11144   }
11145 }
11146 {
11147   \newcommand*\Glsfmttext[1]{%
11148     \Glsxtrtitle{#1}}
11149 }

```

`\glsfmtplural` As above but for the plural value.

```

11150 \ifdef\texorpdfstring
11151 {
11152   \newcommand*\glsfmtplural[1]{%
11153     \texorpdfstring
11154     {\glsxtrtitleplural{#1}}%
11155     {\glsentryplural{#1}}%
11156   }
11157 }
11158 {
11159   \newcommand*\glsfmtplural[1]{%
11160     \glsxtrtitleplural{#1}}
11161 }

```

`\Glsfmtplural` First letter converted to upper case.

```

11162 \ifdef\texorpdfstring
11163 {
11164   \newcommand*\Glsfmtplural[1]{%

```

```

11165 \texorpdfstring
11166 {\Glsxtrtitleplural{#1}}%
11167 {\glsentryplural{#1}}%
11168 }
11169 }
11170 {
11171 \newcommand*{\Glsfmtplural}[1]{%
11172 \Glsxtrtitleplural{#1}}
11173 }

```

`\glsfmtfirst` As above but for the first value.

```

11174 \ifdef\texorpdfstring
11175 {
11176 \newcommand*{\glsfmtfirst}[1]{%
11177 \texorpdfstring
11178 {\glsxtrtitlefirst{#1}}%
11179 {\glsentryfirst{#1}}%
11180 }
11181 }
11182 {
11183 \newcommand*{\glsfmtfirst}[1]{%
11184 \glsxtrtitlefirst{#1}}
11185 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

11186 \ifdef\texorpdfstring
11187 {
11188 \newcommand*{\Glsfmtfirst}[1]{%
11189 \texorpdfstring
11190 {\Glsxtrtitlefirst{#1}}%
11191 {\glsentryfirst{#1}}%
11192 }
11193 }
11194 {
11195 \newcommand*{\Glsfmtfirst}[1]{%
11196 \Glsxtrtitlefirst{#1}}
11197 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

11198 \ifdef\texorpdfstring
11199 {
11200 \newcommand*{\glsfmtfirstpl}[1]{%
11201 \texorpdfstring
11202 {\glsxtrtitlefirstplural{#1}}%
11203 {\glsentryfirstplural{#1}}%
11204 }
11205 }
11206 {
11207 \newcommand*{\glsfmtfirstpl}[1]{%

```

```

11208 \glstrtitlefirstplural{#1}}
11209 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

11210 \ifdef\texorpdfstring
11211 {
11212 \newcommand*\Glsfmtfirstpl}[1]{%
11213 \texorpdfstring
11214 {\Glsxtrtitlefirstplural{#1}}%
11215 {\glstryfirstplural{#1}}%
11216 }
11217 }
11218 {
11219 \newcommand*\Glsfmtfirstpl}[1]{%
11220 \Glsxtrtitlefirstplural{#1}}
11221 }

```

`\glsfmtlong` As above but for the long value.

```

11222 \ifdef\texorpdfstring
11223 {
11224 \newcommand*\glsfmtlong}[1]{%
11225 \texorpdfstring
11226 {\glstrtitlelong{#1}}%
11227 {\glstrylong{#1}}%
11228 }
11229 }
11230 {
11231 \newcommand*\glsfmtlong}[1]{%
11232 \glstrtitlelong{#1}}
11233 }

```

`\Glsfmtlong` First letter converted to upper case.

```

11234 \ifdef\texorpdfstring
11235 {
11236 \newcommand*\Glsfmtlong}[1]{%
11237 \texorpdfstring
11238 {\Glsxtrtitlelong{#1}}%
11239 {\glstrylong{#1}}%
11240 }
11241 }
11242 {
11243 \newcommand*\Glsfmtlong}[1]{%
11244 \Glsxtrtitlelong{#1}}
11245 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

11246 \ifdef\texorpdfstring
11247 {
11248 \newcommand*\glsfmtlongpl}[1]{%

```

```

11249 \texorpdfstring
11250 {\glsxtrtitlelongpl{#1}}%
11251 {\glsentrylongpl{#1}}%
11252 }
11253 }
11254 {
11255 \newcommand*{\glsfmtlongpl}[1]{%
11256 \glsxtrtitlelongpl{#1}}
11257 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

11258 \ifdef\texorpdfstring
11259 {
11260 \newcommand*{\Glsfmtlongpl}[1]{%
11261 \texorpdfstring
11262 {\Glsxtrtitlelongpl{#1}}%
11263 {\glsentrylongpl{#1}}%
11264 }
11265 }
11266 {
11267 \newcommand*{\Glsfmtlongpl}[1]{%
11268 \Glsxtrtitlelongpl{#1}}
11269 }

```

`\glsfmtfull` In-line full format.

```

11270 \ifdef\texorpdfstring
11271 {
11272 \newcommand*{\glsfmtfull}[1]{%
11273 \texorpdfstring
11274 {\glsxtrtitlefull{#1}}%
11275 {\glsxtrinlinefullformat{#1}{}}%
11276 }
11277 }
11278 {
11279 \newcommand*{\glsfmtfull}[1]{%
11280 \glsxtrtitlefull{#1}}
11281 }

```

`\Glsfmtfull` First letter converted to upper case.

```

11282 \ifdef\texorpdfstring
11283 {
11284 \newcommand*{\Glsfmtfull}[1]{%
11285 \texorpdfstring
11286 {\Glsxtrtitlefull{#1}}%
11287 {\Glsxtrinlinefullformat{#1}{}}%
11288 }
11289 }
11290 {
11291 \newcommand*{\Glsfmtfull}[1]{%

```

```

11292 \Glsxtrtitlefull{#1}}
11293 }

```

`\glsfmtfullpl` In-line full plural format.

```

11294 \ifdef\texorpdfstring
11295 {
11296   \newcommand*{\glsfmtfullpl}[1]{%
11297     \texorpdfstring
11298     {\Glsxtrtitlefullpl{#1}}%
11299     {\Glsxtrinlinefullplformat{#1}{}}%
11300   }
11301 }
11302 {
11303   \newcommand*{\glsfmtfullpl}[1]{%
11304     \Glsxtrtitlefullpl{#1}}
11305 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

11306 \ifdef\texorpdfstring
11307 {
11308   \newcommand*{\Glsfmtfullpl}[1]{%
11309     \texorpdfstring
11310     {\Glsxtrtitlefullpl{#1}}%
11311     {\Glsxtrinlinefullplformat{#1}{}}%
11312   }
11313 }
11314 {
11315   \newcommand*{\Glsfmtfullpl}[1]{%
11316     \Glsxtrtitlefullpl{#1}}
11317 }

```

1.9 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

11318 \newcommand*{\RequireGlossariesExtraLang}[1]{%
11319   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
11320 }

```

`sariesExtraLang`

```

11321 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
11322   \ProvidesFile{glossariesxtr-#1.ldf}%
11323 }

```

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.

```

11324 \newcommand{\glxtr@loaddialect}{%
11325   \IfTrackedLanguageFileExists{\this@dialect}%
11326   {glossariesxtr-}% prefix
11327   {.ldf}%
11328   {%
11329     \RequireGlossariesExtraLang{\CurrentTrackedTag}%
11330   }%
11331   {}% not found

```

If `glossaries-extra-bib2gls` has been loaded, `\@glxtrdialecthook` will check for the associated script, otherwise it will do nothing.

```

11332 \@glxtrdialecthook
11333 }

```

```

11334 \@ifpackageloaded{tracklang}
11335 {%
11336   \AnyTrackedLanguages
11337   {%
11338     \ForEachTrackedDialect{\this@dialect}{\glxtr@loaddialect}%
11339   }%
11340   {}%
11341 }
11342 {}

```

Load `glossaries-extra-stylemods` if required.

```

11343 \@glxtr@redefstyles
    and set the style:
11344 \@glxtr@do@style

```

1.10 glossaries-extra-bib2gls.sty

This package provides additional support for `bib2gls` and is automatically loaded by the `record` option.

```

11345 \NeedsTeXFormat{LaTeX2e}
11346 \ProvidesPackage{glossaries-extra-bib2gls}[2018/04/25 v1.30 (NLCT)]

```

These are some convenient macros for use with custom rules.

`\glshex`

```

11347 \newcommand*{\glshex}{\string\u}

```

`\rprovidecommand` For use in `@preamble`, this behaves like `\providecommand` in the document but like `\renewcommand` in `bib2gls`.

```

11348 \newcommand*{\glxtrprovidecommand}{\providecommand}

```

lossarylocation For use with indexcounter and bib2gls.

```
11349 \newcommand*{\glstr@wrglossarylocation}[2]{#1}
```

IndexCounterLink `\GlsXtrIndexCounterLink{<text>}{<label>}`

For use with indexcounter and bib2gls.

```
11350 \ifdef\hyperref
11351 {%
11352   \newcommand*{\GlsXtrIndexCounterLink}[2]{%
11353     \glstrifhasfield{indexcounter}{#2}%
11354     {\hyperref[wrglossary.\glscurrentfieldvalue]{#1}}%
11355     {#1}%
11356   }
11357 }
11358 {
11359   \newcommand*{\GlsXtrIndexCounterLink}[2]{#1}
11360 }
```

\GlsXtrDualField `\GlsXtrDualField`

The internal field used to store the dual label. The dual-field defaults to dual if no value is supplied so that's used as the default.

```
11361 \newcommand*{\GlsXtrDualField}{dual}
```

sXtrDualBackLink `\GlsXtrDualBackLink{<text>}{<label>}`

Adds a hyperlink to the dual entry.

```
11362 \newcommand*{\GlsXtrDualBackLink}[2]{%
11363   \glstrifhasfield{\GlsXtrDualField}{#2}%
11364   {\glshyperlink[#1]{\glscurrentfieldvalue}}%
11365   {#2}%
11366 }
```

TeXEntryAliases Convenient shortcut for use with entry-type-aliases to alias standard BibTeX entry types to @bibtexentry.

```
11367 \newcommand*{\GlsXtrBibTeXEntryAliases}{%
11368   article=bibtexentry,
11369   book=bibtexentry,
11370   booklet=bibtexentry,
11371   conference=bibtexentry,
```



```

11372 inbook=bibtexentry,
11373 incollection=bibtexentry,
11374 inproceedings=bibtexentry,
11375 manual=bibtexentry,
11376 mastersthesis=bibtexentry,
11377 misc=bibtexentry,
11378 phdthesis=bibtexentry,
11379 proceedings=bibtexentry,
11380 techreport=bibtexentry,
11381 unpublished=bibtexentry
11382 }

```

`\provideBibTeXFields` Convenient shortcut to define the standard `BIBTeX` fields.

```

11383 \newcommand*{\GlsXtrProvideBibTeXFields}{%
11384   \glsaddstoragekey{address}{}{\glstrbibaddress}%
11385   \glsaddstoragekey{author}{}{\glstrbibauthor}%
11386   \glsaddstoragekey{booktitle}{}{\glstrbibbooktitle}%
11387   \glsaddstoragekey{chapter}{}{\glstrbibchapter}%
11388   \glsaddstoragekey{edition}{}{\glstrbibedition}%
11389   \glsaddstoragekey{howpublished}{}{\glstrbibhowpublished}%
11390   \glsaddstoragekey{institution}{}{\glstrbibinstitution}%
11391   \glsaddstoragekey{journal}{}{\glstrbibjournal}%
11392   \glsaddstoragekey{month}{}{\glstrbibmonth}%
11393   \glsaddstoragekey{note}{}{\glstrbibnote}%
11394   \glsaddstoragekey{number}{}{\glstrbibnumber}%
11395   \glsaddstoragekey{organization}{}{\glstrbiborganization}%
11396   \glsaddstoragekey{pages}{}{\glstrbibpages}%
11397   \glsaddstoragekey{publisher}{}{\glstrbibpublisher}%
11398   \glsaddstoragekey{school}{}{\glstrbibschooll}%
11399   \glsaddstoragekey{series}{}{\glstrbibseries}%
11400   \glsaddstoragekey{title}{}{\glstrbibtitle}%
11401   \glsaddstoragekey{bibtextype}{}{\glstrbibtype}%
11402   \glsaddstoragekey{volume}{}{\glstrbibvolume}%
11403 }

```

Provide missing Greek letters for use in maths mode. These commands 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 mathematical Greek letters for sorting purposes. The `ℒTeX` version of these commands (provided here) 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`

```
11404 \providecommand*{\Alpha}{\mathrm{A}}
```

`\Beta`

```
11405 \providecommand*{\Beta}{\mathrm{B}}
```

```

\Epsilon
11406 \providecommand*\Epsilon{\mathrm{E}}

\Zeta
11407 \providecommand*\Zeta{\mathrm{Z}}

\Eta
11408 \providecommand*\Eta{\mathrm{H}}

\Iota
11409 \providecommand*\Iota{\mathrm{I}}

\Kappa
11410 \providecommand*\Kappa{\mathrm{K}}

\Mu
11411 \providecommand*\Mu{\mathrm{M}}

\Nu
11412 \providecommand*\Nu{\mathrm{N}}

\Omicron
11413 \providecommand*\Omicron{\mathrm{O}}

\Rho
11414 \providecommand*\Rho{\mathrm{P}}

\Tau
11415 \providecommand*\Tau{\mathrm{T}}

\Chi
11416 \providecommand*\Chi{\mathrm{X}}

\Digamma
11417 \providecommand*\Digamma{\mathrm{F}}

\omicron
11418 \providecommand*\omicron{\mathit{o}}

    Provide corresponding upright characters if upgreek has been loaded. (The upper case
    characters are the same as above.)
11419 \@ifpackageloaded{upgreek}%
11420 {

\Upalpha
11421 \providecommand*\Upalpha{\mathrm{A}}

```

```

\Upbeta
11422 \providecommand*\Upbeta{\mathrm{B}}

\Upsilon
11423 \providecommand*\Upsilon{\mathrm{E}}

\Upzeta
11424 \providecommand*\Upzeta{\mathrm{Z}}

\Upeta
11425 \providecommand*\Upeta{\mathrm{H}}

\Upiota
11426 \providecommand*\Upiota{\mathrm{I}}

\Upkappa
11427 \providecommand*\Upkappa{\mathrm{K}}

\Upmu
11428 \providecommand*\Upmu{\mathrm{M}}

\Upnu
11429 \providecommand*\Upnu{\mathrm{N}}

\Upomicron
11430 \providecommand*\Upomicron{\mathrm{O}}

\Uprho
11431 \providecommand*\Uprho{\mathrm{P}}

\Uptau
11432 \providecommand*\Uptau{\mathrm{T}}

\Upchi
11433 \providecommand*\Upchi{\mathrm{X}}

\upomicron
11434 \providecommand*\upomicron{\mathrm{o}}

11435 }%
11436 {}% upgreek.sty not loaded

```

This package provides some basic rules, but it's 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 or ISO code. 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. These control characters are unlikely to appear in any entry fields but are provided for completeness. `\string` is used for punctuation characters in case they’ve been made active.

```
11437 \newcommand*{\glxtrcontrolrules}{%
11438 \string'\glshex 200B\string'\string=\glshex 200C\string=\glshex 200D
11439 \string=\glshex 200E\string=\glshex 200F\string=\glshex 0000\string=\glshex 0001
11440 \string=\glshex 0002\string=\glshex 0003\string=\glshex 0004\string=\glshex 0005
11441 \string=\glshex 0006\string=\glshex 0007\string=\glshex 0008
11442 \string=\string'\glshex 0009\string'\string=\string'\glshex 000B\string'
11443 \string=\glshex 000E\string=\glshex 000F\string=\string'\glshex
11444 0010\string'\string=\glshex 0011
11445 \string=\glshex 0012\string=\glshex 0013\string=\glshex 0014\string=\glshex 0015
11446 \string=\glshex 0016\string=\glshex 0017\string=\glshex 0018\string=\glshex 0019
11447 \string=\glshex 001A\string=\glshex 001B\string=\glshex 001C\string=\glshex 001D
11448 \string=\glshex 001E\string=\glshex 001F\string=\glshex 007F\string=\glshex 0080
11449 \string=\glshex 0081\string=\glshex 0082\string=\glshex 0083\string=\glshex 0084
11450 \string=\glshex 0085\string=\glshex 0086\string=\glshex 0087\string=\glshex 0088
11451 \string=\glshex 0089\string=\glshex 008A\string=\glshex 008B\string=\glshex 008C
11452 \string=\glshex 008D\string=\glshex 008E\string=\glshex 008F\string=\glshex 0090
11453 \string=\glshex 0091\string=\glshex 0092\string=\glshex 0093\string=\glshex 0094
11454 \string=\glshex 0095\string=\glshex 0096\string=\glshex 0097\string=\glshex 0098
11455 \string=\glshex 0099\string=\glshex 009A\string=\glshex 009B\string=\glshex 009C
11456 \string=\glshex 009D\string=\glshex 009E\string=\glshex 009F
11457 }
```

`lsxtrspacerules` These are space characters.

```

11458 \newcommand*{\glxtrspacerules}{%
11459 \string' \string'\string;
11460 \string'\glshex 00A0\string'\string;
11461 \string'\glshex 2000\string'\string;
11462 \string'\glshex 2001\string'\string;
11463 \string'\glshex 2002\string'\string;
11464 \string'\glshex 2003\string'\string;
11465 \string'\glshex 2004\string'\string;
11466 \string'\glshex 2005\string'\string;
11467 \string'\glshex 2006\string'\string;
11468 \string'\glshex 2007\string'\string;
11469 \string'\glshex 2008\string'\string;
11470 \string'\glshex 2009\string'\string;
11471 \string'\glshex 200A\string'\string;
11472 \string'\glshex 3000\string'
11473 }

```

nprintablerules These are non-printable characters (BOM, tabs, line feed and carriage return).

```

11474 \newcommand*{\glxtrnonprintablerules}{%
11475 \string'\glshex FEFF\string'\string;
11476 \string'\glshex 000A\string'\string;
11477 \string'\glshex 0009\string'\string;
11478 \string'\glshex 000C\string'\string;
11479 \string'\glshex 000B\string'
11480 }

```

gdiacriticrules Combining diacritic marks. This is split into multiple macros.

```

11481 \newcommand*{\glxtrcombiningdiacriticrules}{%
11482 \glxtrcombiningdiacriticIrules\string;
11483 \glxtrcombiningdiacriticIIrules\string;
11484 \glxtrcombiningdiacriticIIIrules\string;
11485 \glxtrcombiningdiacriticIVrules
11486 }

```

diacriticIrules First set of combining diacritic marks.

```

11487 \newcommand*{\glxtrcombiningdiacriticIrules}{%
11488 \glshex 0301\string;% combining acute
11489 \glshex 0300\string;% combining grave
11490 \glshex 0306\string;% combining breve
11491 \glshex 0302\string;% combining circumflex
11492 \glshex 030C\string;% combining caron
11493 \glshex 030A\string;% combining ring
11494 \glshex 030D\string;% combining vertical line above
11495 \glshex 0308\string;% combining diaeresis
11496 \glshex 030B\string;% combining double acute
11497 \glshex 0303\string;% combining tilde
11498 \glshex 0307\string;% combining dot above
11499 \glshex 0304% combining macron
11500 }

```

iacriticIIrules Second set of combining diacritic marks.

```
11501 \newcommand*{\glxtrcombingdiacriticIIrules}{%
11502 \glshex 0337\string;% combining short solidus overlay
11503 \glshex 0327\string;% combining cedilla
11504 \glshex 0328\string;% combining ogonek
11505 \glshex 0323\string;% combining dot below
11506 \glshex 0332\string;% combining low line
11507 \glshex 0305\string;% combining overline
11508 \glshex 0309\string;% combining hook above
11509 \glshex 030E\string;% combining double vertical line above
11510 \glshex 030F\string;% combining double grave accent
11511 \glshex 0310\string;% combining candrabindu
11512 \glshex 0311\string;% combining inverted breve
11513 \glshex 0312\string;% combining turned comma above
11514 \glshex 0313\string;% combining comma above
11515 \glshex 0314\string;% combining reversed comma above
11516 \glshex 0315\string;% combining comma above right
11517 \glshex 0316\string;% combining grave accent below
11518 \glshex 0317% combining acute accent below
11519 }
```

acriticIIIrules Third set of combining diacritic marks.

```
11520 \newcommand*{\glxtrcombingdiacriticIIIrules}{%
11521 \glshex 0318\string;% combining left tack below
11522 \glshex 0319\string;% combining right tack below
11523 \glshex 031A\string;% combining left angle above
11524 \glshex 031B\string;% combining horn
11525 \glshex 031C\string;% combining left half ring below
11526 \glshex 031D\string;% combining up tack below
11527 \glshex 031E\string;% combining down tack below
11528 \glshex 031F\string;% combining plus sign below
11529 \glshex 0320\string;% combining minus sign below
11530 \glshex 0321\string;% combining palatalized hook below
11531 \glshex 0322\string;% combining retroflex hook below
11532 \glshex 0324\string;% combining diaeresis below
11533 \glshex 0325\string;% combining ring below
11534 \glshex 0326\string;% combining comma below
11535 \glshex 0329\string;% combining vertical line below
11536 \glshex 032A\string;% combining bridge below
11537 \glshex 032B\string;% combining inverted double arch below
11538 \glshex 032C\string;% combining caron below
11539 \glshex 032D\string;% combining circumflex accent below
11540 \glshex 032E\string;% combining breve below
11541 \glshex 032F\string;% combining inverted breve below
11542 \glshex 0330\string;% combining tilde below
11543 \glshex 0331\string;% combining macron below
11544 \glshex 0333\string;% combining double low line
11545 \glshex 0334\string;% combining tilde overlay
11546 \glshex 0335\string;% combining short stroke overlay
```

```

11547 \glshex 0336\string;% combining long stroke overlay
11548 \glshex 0338\string;% combining long solidus overlay
11549 \glshex 0339\string;% combining combining right half ring below
11550 \glshex 033A\string;% combining inverted bridge below
11551 \glshex 033B\string;% combining square below
11552 \glshex 033C\string;% combining seagull below
11553 \glshex 033D\string;% combining x above
11554 \glshex 033E\string;% combining vertical tilde
11555 \glshex 033F\string;% combining double overline
11556 \glshex 0342\string;% combining Greek perispomeni
11557 \glshex 0344\string;% combining Greek dialytika tonos
11558 \glshex 0345\string;% combining Greek ypogegrammeni
11559 \glshex 0360\string;% combining double tilde
11560 \glshex 0361\string;% combining double inverted breve
11561 \glshex 0483\string;% combining Cyrillic titlo
11562 \glshex 0484\string;% combining Cyrillic palatalization
11563 \glshex 0485\string;% combining Cyrillic dasia pneumata
11564 \glshex 0486% combining Cyrillic psili pneumata
11565 }

```

iacriticIVrules Fourth set of combining diacritic marks.

```

11566 \newcommand*{\glxtrcombiningdiacriticIVrules}{%
11567 \glshex 20D0\string;% combining left harpoon above
11568 \glshex 20D1\string;% combining right harpoon above
11569 \glshex 20D2\string;% combining long vertical line overlay
11570 \glshex 20D3\string;% combining short vertical line overlay
11571 \glshex 20D4\string;% combining anticlockwise arrow above
11572 \glshex 20D5\string;% combining clockwise arrow above
11573 \glshex 20D6\string;% combining left arrow above
11574 \glshex 20D7\string;% combining right arrow above
11575 \glshex 20D8\string;% combining ring overlay
11576 \glshex 20D9\string;% combining clockwise ring overlay
11577 \glshex 20DA\string;% combining anticlockwise ring overlay
11578 \glshex 20DB\string;% combining three dots above
11579 \glshex 20DC\string;% combining four dots above
11580 \glshex 20DD\string;% combining enclosing circle
11581 \glshex 20DE\string;% combining enclosing square
11582 \glshex 20DF\string;% combining enclosing diamond
11583 \glshex 20E0\string;% combining enclosing circle backslash
11584 \glshex 20E1% combining left right arrow above
11585 }

```

sxtrhyphenrules Hyphens.

```

11586 \newcommand*{\glxtrhyphenrules}{%
11587 \string'\string-\string'\string;% ASCII hyphen
11588 \glshex 00AD\string;% soft hyphen
11589 \glshex 2010\string;% hyphen
11590 \glshex 2011\string;% non-breaking hyphen
11591 \glshex 2012\string;% figure dash

```

```

11592 \glshex 2013\string;% en dash
11593 \glshex 2014\string;% em dash
11594 \glshex 2015\string;% horizontal bar
11595 \glshex 2212\string=\glshex 207B\string=\glshex 208B% minus sign
11596 }

```

generalpuncrules General punctuation.

```

11597 \newcommand*{\glxtrgeneralpuncrules}{%
11598   \glxtrgeneralpuncIrules
11599   \string<\glxtrcurrencyrules
11600   \string<\glxtrgeneralpuncIIrules
11601 }

```

generalpuncIrules First set of general punctuation.

```

11602 \newcommand*{\glxtrgeneralpuncIrules}{%
11603   \string'\glshex 005F\string'% underscore
11604   \string<\glshex 00AF% macron
11605   \string<\string'\glshex 002C\string'% comma
11606   \string<\string'\glshex 003B\string'% semi-colon
11607   \string<\string'\glshex 003A\string'% colon
11608   \string<\string'\glshex 0021\string'% exclamation mark
11609   \string<\glshex 00A1% inverted exclamation mark
11610   \string<\string'\glshex 003F\string'% question mark
11611   \string<\glshex 00BF% inverted question mark
11612   \string<\string'\glshex 002F\string'% solidus
11613   \string<\string'\glshex 002E\string'% full stop
11614   \string<\glshex 00B4% acute accent
11615   \string<\string'\glshex 0060\string'% grave accent
11616   \string<\string'\glshex 005E\string'% circumflex accent
11617   \string<\glshex 00A8% diaeresis
11618   \string<\string'\glshex 007E\string'% tilde
11619   \string<\glshex 00B7% middle dot
11620   \string<\glshex 00B8% cedilla
11621   \string<\string'\glshex 0027\string'% straight apostrophe
11622   \string<\string'\glshex 0022\string'% straight double quote
11623   \string<\glshex 00AB% left guillemet
11624   \string<\glshex 00BB% right guillemet
11625   \string<\string'\glshex 0028\string'% left parenthesis
11626   \string=\glshex 207D\string=\glshex 208D% super/subscript left parenthesis
11627   \string<\string'\glshex 0029\string'% right parenthesis
11628   \string=\glshex 207E\string=\glshex 208E% super/subscript right parenthesis
11629   \string<\string'\glshex 005B\string'% left square bracket
11630   \string<\string'\glshex 005D\string'% right square bracket
11631   \string<\string'\glshex 007B\string'% left curly bracket
11632   \string<\string'\glshex 007D\string'% right curly bracket
11633   \string<\glshex 00A7% section sign
11634   \string<\glshex 00B6% pilcrow sign
11635   \string<\glshex 00A9% copyright sign
11636   \string<\glshex 00AE% registered sign

```



```

11637 \string<\string'\glshex 0040\string'% at sign
11638 }

```

trcurrencyrules General punctuation.

```

11639 \newcommand*{\glxtrcurrencyrules}{%
11640 \glshex 00A4% currency sign
11641 \string<\glshex 0E3F% Thai currency symbol baht
11642 \string<\glshex 00A2% cent sign
11643 \string<\glshex 20A1% colon sign
11644 \string<\glshex 20A2% cruzeiro sign
11645 \string<\string'\glshex 0024\string'% dollar sign
11646 \string<\glshex 20AB% dong sign
11647 \string<\glshex 20AC% euro sign
11648 \string<\glshex 20A3% French franc sign
11649 \string<\glshex 20A4% lira sign
11650 \string<\glshex 20A5% mill sign
11651 \string<\glshex 20A6% naira sign
11652 \string<\glshex 20A7% peseta sign
11653 \string<\glshex 00A3% pound sign
11654 \string<\glshex 20A8% rupee sign
11655 \string<\glshex 20AA% new sheqel sign
11656 \string<\glshex 20A9% won sign
11657 \string<\glshex 00A5% yen sign
11658 }

```

eralpuncIIrules Second set of general punctuation.

```

11659 \newcommand*{\glxtrgeneralpuncIIrules}{%
11660 \string'\glshex 002A\string'% asterisk
11661 \string<\string'\glshex 005C\string'% backslash
11662 \string<\string'\glshex 0026\string'% ampersand
11663 \string<\string'\glshex 0023\string'% hash sign
11664 \string<\string'\glshex 0025\string'% percent sign
11665 \string<\string'\glshex 002B\string'% plus sign
11666 \string=>\glshex 207A\string=>\glshex 208A% super/subscript plus sign
11667 \string<\glshex 00B1% plus-minus sign
11668 \string<\glshex 00F7% division sign
11669 \string<\glshex 00D7% multiplication sign
11670 \string<\string'\glshex 003C\string'% less-than sign
11671 \string<\string'\glshex 003D\string'% equals sign
11672 \string<\string'\glshex 003E\string'% greater-than sign
11673 \string<\glshex 00AC% not sign
11674 \string<\string'\glshex 007C\string'% vertical bar (pipe)
11675 \string<\glshex 00A6% broken bar
11676 \string<\glshex 00B0% degree sign
11677 \string<\glshex 00B5% micron sign
11678 }

```

eralLatinIrules Basic Latin alphabet.

```

11679 \newcommand*{\glxtrGeneralLatinIrules}{%

```

```

11680 \glxtrLatinA
11681 \string<b,B%
11682 \string<c,C%
11683 \string<d,D%
11684 \string<\glxtrLatinE
11685 \string<f,F%
11686 \string<g,G%
11687 \string<\glxtrLatinH
11688 \string<\glxtrLatinI
11689 \string<j,J%
11690 \string<\glxtrLatinK
11691 \string<\glxtrLatinL
11692 \string<\glxtrLatinM
11693 \string<\glxtrLatinN
11694 \string<\glxtrLatinO
11695 \string<\glxtrLatinP
11696 \string<q,Q%
11697 \string<r,R%
11698 \string<\glxtrLatinS
11699 \string<\glxtrLatinT
11700 \string<u,U%
11701 \string<v,V%
11702 \string<w,W%
11703 \string<\glxtrLatinX
11704 \string<y,Y%
11705 \string<z,Z
11706 }

```

raLatinIIrules General Latin alphabet (eth between D and E, ß treated as SS).

```

11707 \newcommand*{\glxtrGeneralLatinIIrules}{%
11708 \glxtrLatinA
11709 \string<b,B%
11710 \string<c,C%
11711 \string<d,D%
11712 \string<\glxtrLatinEth
11713 \string<\glxtrLatinE
11714 \string<f,F%
11715 \string<g,G%
11716 \string<\glxtrLatinH
11717 \string<\glxtrLatinI
11718 \string<j,J%
11719 \string<\glxtrLatinK
11720 \string<\glxtrLatinL
11721 \string<\glxtrLatinM
11722 \string<\glxtrLatinN
11723 \string<\glxtrLatinO
11724 \string<\glxtrLatinP
11725 \string<q,Q%
11726 \string<r,R%

```

```

11727 \string<\glxtrLatinS
11728 \string& SS \string, \glxtrLatinEszettSs
11729 \string<\glxtrLatinT
11730 \string<u,U%
11731 \string<v,V%
11732 \string<w,W%
11733 \string<\glxtrLatinX
11734 \string<y,Y%
11735 \string<z,Z%
11736 }

```

allLatinIIIrules General Latin alphabet (eth between D and E, ß treated as SZ).

```

11737 \newcommand*{\glxtrGeneralLatinIIIrules}{%
11738 \glxtrLatinA
11739 \string<b,B%
11740 \string<c,C%
11741 \string<d,D%
11742 \string<\glxtrLatinEth
11743 \string<\glxtrLatinE
11744 \string<f,F%
11745 \string<g,G%
11746 \string<\glxtrLatinH
11747 \string<\glxtrLatinI
11748 \string<j,J%
11749 \string<\glxtrLatinK
11750 \string<\glxtrLatinL
11751 \string<\glxtrLatinM
11752 \string<\glxtrLatinN
11753 \string<\glxtrLatinO
11754 \string<\glxtrLatinP
11755 \string<q,Q%
11756 \string<r,R%
11757 \string<\glxtrLatinS
11758 \string& SZ, \glxtrLatinEszettSz
11759 \string<\glxtrLatinT
11760 \string<u,U%
11761 \string<v,V%
11762 \string<w,W%
11763 \string<\glxtrLatinX
11764 \string<y,Y%
11765 \string<z,Z%
11766 }

```

ralLatinIVrules General Latin alphabet (Æ treated as AE and Ætreated as OE, ptreated as TH, ß treated as SS, eth between D and E).

```

11767 \newcommand*{\glxtrGeneralLatinIVrules}{%
11768 \glxtrLatinA
11769 \string& AE , \glxtrLatinAELigature
11770 \string<b,B%

```

```

11771 \string<c,C%
11772 \string<d,D%
11773 \string<\glxtrLatinEth
11774 \string<\glxtrLatinE
11775 \string<f,F%
11776 \string<g,G%
11777 \string<\glxtrLatinH
11778 \string<\glxtrLatinI
11779 \string<j,J%
11780 \string<\glxtrLatinK
11781 \string<\glxtrLatinL
11782 \string<\glxtrLatinM
11783 \string<\glxtrLatinN
11784 \string<\glxtrLatinO
11785 \string& OE , \glxtrLatinOELigature
11786 \string<\glxtrLatinP
11787 \string<q,Q%
11788 \string<r,R%
11789 \string<\glxtrLatinS
11790 \string& SS , \glxtrLatinEszettSs
11791 \string<\glxtrLatinT
11792 \string& th =\glshex 00DE
11793 \string& TH =\glshex 00FE
11794 \string<u,U%
11795 \string<v,V%
11796 \string<w,W%
11797 \string<\glxtrLatinX
11798 \string<y,Y%
11799 \string<z,Z%
11800 }

```

eralLatinVrules General Latin alphabet (eth between D and E, ß treated as SS, Þ treated as TH).

```

11801 \newcommand*{\glxtrGeneralLatinVrules}{%
11802 \glxtrLatinA
11803 \string<b,B%
11804 \string<c,C%
11805 \string<d,D%
11806 \string<\glxtrLatinEth
11807 \string<\glxtrLatinE
11808 \string<f,F%
11809 \string<g,G%
11810 \string<\glxtrLatinH
11811 \string<\glxtrLatinI
11812 \string<j,J%
11813 \string<\glxtrLatinK
11814 \string<\glxtrLatinL
11815 \string<\glxtrLatinM
11816 \string<\glxtrLatinN
11817 \string<\glxtrLatinO

```

```

11818 \string<\glxtrLatinP
11819 \string<q,Q%
11820 \string<r,R%
11821 \string<\glxtrLatinS
11822 \string& SS , \glxtrLatinEszettSs
11823 \string<\glxtrLatinT
11824 \string& th =\glshex 00DE
11825 \string& TH =\glshex 00FE
11826 \string<u,U%
11827 \string<v,V%
11828 \string<w,W%
11829 \string<\glxtrLatinX
11830 \string<y,Y%
11831 \string<z,Z%
11832 }

```

raLatinVrules General Latin alphabet (eth between D and E, ß treated as SZ, Þ treated as TH).

```

11833 \newcommand*{\glxtrGeneralLatinVrules}{%
11834 \glxtrLatinA
11835 \string<b,B%
11836 \string<c,C%
11837 \string<d,D%
11838 \string<\glxtrLatinEth
11839 \string<\glxtrLatinE
11840 \string<f,F%
11841 \string<g,G%
11842 \string<\glxtrLatinH
11843 \string<\glxtrLatinI
11844 \string<j,J%
11845 \string<\glxtrLatinK
11846 \string<\glxtrLatinL
11847 \string<\glxtrLatinM
11848 \string<\glxtrLatinN
11849 \string<\glxtrLatinO
11850 \string<\glxtrLatinP
11851 \string<q,Q%
11852 \string<r,R%
11853 \string<\glxtrLatinS
11854 \string& SZ , \glxtrLatinEszettSz
11855 \string<\glxtrLatinT
11856 \string& th =\glshex 00DE
11857 \string& TH =\glshex 00FE
11858 \string<u,U%
11859 \string<v,V%
11860 \string<w,W%
11861 \string<\glxtrLatinX
11862 \string<y,Y%
11863 \string<z,Z%
11864 }

```

alLatinVIIrules General Latin alphabet (Æ between A and B, eth between D and E, insular G as G, Ć between O and P, long S equivalent to S, Þ between T and U and wynn as W).

```

11865 \newcommand*{\glxtrGeneralLatinVIIrules}{%
11866 \glxtrLatinA
11867 \string<\glxtrLatinAELigature
11868 \string<b,B%
11869 \string<c,C%
11870 \string<d,D%
11871 \string<\glxtrLatinEth
11872 \string<\glxtrLatinE
11873 \string<f,F%
11874 \string<\glxtrLatinInsularG
11875 \string<\glxtrLatinH
11876 \string<\glxtrLatinI
11877 \string<j,J%
11878 \string<\glxtrLatinK
11879 \string<\glxtrLatinL
11880 \string<\glxtrLatinM
11881 \string<\glxtrLatinN
11882 \string<\glxtrLatinO
11883 \string<\glxtrLatinOELigature
11884 \string<\glxtrLatinP
11885 \string<q,Q%
11886 \string<r,R%
11887 \string<\glshex 017F=\glxtrLatinS % s and long s
11888 \string<\glxtrLatinT
11889 \string<\glxtrLatinThorn
11890 \string<u,U%
11891 \string<v,V%
11892 \string< w\string=\glshex 01BF, W\string=\glshex 01F7
11893 \string<\glxtrLatinX
11894 \string<y,Y%
11895 \string<z,Z%
11896 }

```

lLatinVIIIrules General Latin alphabet (Æ treated as AE and Ć treated as OE, Þ treated as TH, ß treated as SS, eth treated as D, Ø treated as O, Ł treated as L).

```

11897 \newcommand*{\glxtrGeneralLatinVIIIrules}{%
11898 \glxtrLatinA
11899 \string& AE , \glxtrLatinAELigature
11900 \string<b,B%
11901 \string<c,C%
11902 \string<\glshex 00F0\string;d,\glshex 00D0\string;D% D and eth
11903 \string<\glxtrLatinE
11904 \string<f,F%
11905 \string<g,G%
11906 \string<\glxtrLatinH
11907 \string<\glxtrLatinI
11908 \string<j,J%

```

```

11909 \string<\glxtrLatinK
11910 \string<\glshex 0142\string=\glxtrLatinL\string=\glshex 0141% L and \L
11911 \string<\glxtrLatinM
11912 \string<\glxtrLatinN
11913 \string<\glshex 00F8\string=\glxtrLatinO\string=\glshex 00D8% O and \O
11914 \string& OE , \glxtrLatinOELigature
11915 \string<\glxtrLatinP
11916 \string<q,Q%
11917 \string<r,R%
11918 \string<\glxtrLatinS
11919 \string& SS , \glxtrLatinEszettSs
11920 \string<\glxtrLatinT
11921 \string& th =\glshex 00DE
11922 \string& TH =\glshex 00FE
11923 \string<u,U%
11924 \string<v,V%
11925 \string<w,W%
11926 \string<\glxtrLatinX
11927 \string<y,Y%
11928 \string<z,Z%
11929 }

```

\glxtrLatinA

```

11930 \newcommand*{\glxtrLatinA}{%
11931   a\string=\glshex 00AA\string=\glshex 2090,A
11932 }

```

\glxtrLatinE

```

11933 \newcommand*{\glxtrLatinE}{%
11934   e\string=\glshex 2091,E
11935 }

```

\glxtrLatinH

```

11936 \newcommand*{\glxtrLatinH}{%
11937   h\string=\glshex 2095,H
11938 }

```

\glxtrLatinI

```

11939 \newcommand*{\glxtrLatinI}{%
11940   i\string=\glshex 2071,I
11941 }

```

\glxtrLatinK

```

11942 \newcommand*{\glxtrLatinK}{%
11943   k\string=\glshex 2096,K
11944 }

```

\glxtrLatinL

```

11945 \newcommand*{\glxtrLatinL}{%
11946   l\string=\glshex 2097,L
11947 }

```

\glxtrLatinM

```

11948 \newcommand*{\glxtrLatinM}{%
11949   m\string=\glshex 2098,M
11950 }

```

\glxtrLatinN

```

11951 \newcommand*{\glxtrLatinN}{%
11952   n\string=\glshex 207F\string=\glshex 2099,N
11953 }

```

\glxtrLatinO

```

11954 \newcommand*{\glxtrLatinO}{%
11955   o\string=\glshex 00BA\string=\glshex 2092,O
11956 }

```

\glxtrLatinP

```

11957 \newcommand*{\glxtrLatinP}{%
11958   p\string=\glshex 209A,P
11959 }

```

\glxtrLatinS

```

11960 \newcommand*{\glxtrLatinS}{%
11961   s\string=\glshex 209B,S
11962 }

```

\glxtrLatinT

```

11963 \newcommand*{\glxtrLatinT}{%
11964   t\string=\glshex 209C,T
11965 }

```

\glxtrLatinX

```

11966 \newcommand*{\glxtrLatinX}{%
11967   x\string=\glshex 2093,X
11968 }

```

\glxtrLatinSchwa Latin schwa (lower case, subscript and upper case).

```

11969 \newcommand*{\glxtrLatinSchwa}{%
11970   \glshex 0259\string=\glshex 2094,\glshex 018F
11971 }

```

\glxtrLatinEszettSs

```

11972 \newcommand*{\glxtrLatinEszettSs}{%
11973   \glshex 00DF% eszett
11974   \string=\glshex 017Fs % long S s
11975 }

```


trLatinEszettSz

```
11976 \newcommand*{\glxtrLatinEszettSz}{%
11977   \glshex 00DF% eszett
11978   \string= \glshex 017Fz % long S z
11979 }
```

\glxtrLatinEth

```
11980 \newcommand*{\glxtrLatinEth}{%
11981   \glshex 00F0,\glshex 00D0% eth
11982 }
```

lsxtrLatinThorn

```
11983 \newcommand*{\glxtrLatinThorn}{%
11984   \glshex 00FE,\glshex 00DE% thorn
11985 }
```

LatinAELigature

```
11986 \newcommand*{\glxtrLatinAELigature}{%
11987   \glshex 00E6,\glshex 00C6% AE-ligature
11988 }
```

LatinOELigature

```
11989 \newcommand*{\glxtrLatinOELigature}{%
11990   \glshex 0153,\glshex 0152% OE-ligature
11991 }
```

\glxtrLatinAA

```
11992 \newcommand*{\glxtrLatinAA}{%
11993   \glshex 00E5=a\glshex 030A,% \aa
11994   \glshex 00C5=A\glshex 030A% \AA
11995 }
```

glxtrLatinWynn

```
11996 \newcommand*{\glxtrLatinWynn}{%
11997   \glshex 01BF,\glshex 01F7% wynn
11998 }
```

trLatinInsularG

```
11999 \newcommand*{\glxtrLatinInsularG}{%
12000   \glshex 1D79,\glshex A77D% insular G
12001   \string; g, G
12002 }
```

sxtrLatinOslash

```
12003 \newcommand*{\glxtrLatinOslash}{%
12004   \glshex 00F8,\glshex 00D8% \o, \O
12005 }
```

sxtrLatinLslash

```
12006 \newcommand*{\glxtrLatinLslash}{%
12007   \glshex 0142,\glshex 0141% \l, \L
12008 }
```

thUpGreekIrules Includes digamma between epsilon and zeta.

```
12009 \newcommand*{\glxtrMathUpGreekIrules}{%
12010   \glxtrUpAlpha
12011   \string<\glxtrUpBeta
12012   \string<\glxtrUpGamma
12013   \string<\glxtrUpDelta
12014   \string<\glxtrUpEpsilon
12015   \string<\glxtrUpDigamma
12016   \string<\glxtrUpZeta
12017   \string<\glxtrUpEta
12018   \string<\glxtrUpTheta
12019   \string<\glxtrUpIota
12020   \string<\glxtrUpKappa
12021   \string<\glxtrUpLambda
12022   \string<\glxtrUpMu
12023   \string<\glxtrUpNu
12024   \string<\glxtrUpXi
12025   \string<\glxtrUpOmicron
12026   \string<\glxtrUpPi
12027   \string<\glxtrUpRho
12028   \string<\glxtrUpSigma
12029   \string<\glxtrUpTau
12030   \string<\glxtrUpUpsilon
12031   \string<\glxtrUpPhi
12032   \string<\glxtrUpChi
12033   \string<\glxtrUpPsi
12034   \string<\glxtrUpOmega
12035 }
```

hUpGreekIIrules Doesn't include digamma.

```
12036 \newcommand*{\glxtrMathUpGreekIIrules}{%
12037   \glxtrUpAlpha
12038   \string<\glxtrUpBeta
12039   \string<\glxtrUpGamma
12040   \string<\glxtrUpDelta
12041   \string<\glxtrUpEpsilon
12042   \string<\glxtrUpZeta
12043   \string<\glxtrUpEta
12044   \string<\glxtrUpTheta
12045   \string<\glxtrUpIota
12046   \string<\glxtrUpKappa
12047   \string<\glxtrUpLambda
12048   \string<\glxtrUpMu
12049   \string<\glxtrUpNu
```

```

12050 \string<\glxtrUpXi
12051 \string<\glxtrUpOmicron
12052 \string<\glxtrUpPi
12053 \string<\glxtrUpRho
12054 \string<\glxtrUpSigma
12055 \string<\glxtrUpTau
12056 \string<\glxtrUpUpsilon
12057 \string<\glxtrUpPhi
12058 \string<\glxtrUpChi
12059 \string<\glxtrUpPsi
12060 \string<\glxtrUpOmega
12061 }

```

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.

```

12062 \newcommand*{\glxtrMathItalicGreekIrules}{%
12063 \glxtrMathItalicAlpha
12064 \string<\glxtrMathItalicBeta
12065 \string<\glxtrMathItalicGamma
12066 \string<\glxtrMathItalicDelta
12067 \string<\glxtrMathItalicEpsilon
12068 \string<\glxtrUpDigamma
12069 \string<\glxtrMathItalicZeta
12070 \string<\glxtrMathItalicEta
12071 \string<\glxtrMathItalicTheta
12072 \string<\glxtrMathItalicIota
12073 \string<\glxtrMathItalicKappa
12074 \string<\glxtrMathItalicLambda
12075 \string<\glxtrMathItalicMu
12076 \string<\glxtrMathItalicNu
12077 \string<\glxtrMathItalicXi
12078 \string<\glxtrMathItalicOmicron
12079 \string<\glxtrMathItalicPi
12080 \string<\glxtrMathItalicRho
12081 \string<\glxtrMathItalicSigma
12082 \string<\glxtrMathItalicTau
12083 \string<\glxtrMathItalicUpsilon
12084 \string<\glxtrMathItalicPhi
12085 \string<\glxtrMathItalicChi
12086 \string<\glxtrMathItalicPsi
12087 \string<\glxtrMathItalicOmega
12088 }

```

alicGreekIIrules Doesn't include digamma.

```

12089 \newcommand*{\glxtrMathItalicGreekIIrules}{%
12090 \glxtrMathItalicAlpha
12091 \string<\glxtrMathItalicBeta
12092 \string<\glxtrMathItalicGamma
12093 \string<\glxtrMathItalicDelta

```

```

12094 \string<\glxtrMathItalicEpsilon
12095 \string<\glxtrMathItalicZeta
12096 \string<\glxtrMathItalicEta
12097 \string<\glxtrMathItalicTheta
12098 \string<\glxtrMathItalicIota
12099 \string<\glxtrMathItalicKappa
12100 \string<\glxtrMathItalicLambda
12101 \string<\glxtrMathItalicMu
12102 \string<\glxtrMathItalicNu
12103 \string<\glxtrMathItalicXi
12104 \string<\glxtrMathItalicOmicron
12105 \string<\glxtrMathItalicPi
12106 \string<\glxtrMathItalicRho
12107 \string<\glxtrMathItalicSigma
12108 \string<\glxtrMathItalicTau
12109 \string<\glxtrMathItalicUpsilon
12110 \string<\glxtrMathItalicPhi
12111 \string<\glxtrMathItalicChi
12112 \string<\glxtrMathItalicPsi
12113 \string<\glxtrMathItalicOmega
12114 }

```

UpperGreekIrules Upper case only (includes upright digamma).

```

12115 \newcommand*{\glxtrMathItalicUpperGreekIrules}{%
12116 \glshex 1D6E2% upper case alpha (maths italic)
12117 \string<\glshex 1D6E3% upper case beta (maths italic)
12118 \string<\glshex 1D6E4% upper case gamma (maths italic)
12119 \string<\glshex 1D6E5% upper case delta (maths italic)
12120 \string<\glshex 1D6E6% upper case epsilon (maths italic)
12121 \string<\glshex 03DC% upper case digamma
12122 \string<\glshex 1D6E7% upper case zeta (maths italic)
12123 \string<\glshex 1D6E8% upper case eta (maths italic)
12124 \string<\glshex 1D6E9% upper case theta (maths italic)
12125 \string<\glshex 1D6F3% upper case theta variant (maths italic)
12126 \string<\glshex 1D6EA% upper case iota (maths italic)
12127 \string<\glshex 1D6EB% upper case kappa (maths italic)
12128 \string<\glshex 1D6EC% upper case lambda (maths italic)
12129 \string<\glshex 1D6ED% upper case mu (maths italic)
12130 \string<\glshex 1D6EE% upper case nu (maths italic)
12131 \string<\glshex 1D6EF% upper case xi (maths italic)
12132 \string<\glshex 1D6F0% upper case omicron (maths italic)
12133 \string<\glshex 1D6F1% upper case pi (maths italic)
12134 \string<\glshex 1D6F2% upper case rho (maths italic)
12135 \string<\glshex 1D6F4% upper case sigma (maths italic)
12136 \string<\glshex 1D6F5% upper case tau (maths italic)
12137 \string<\glshex 1D6F6% upper case upsilon (maths italic)
12138 \string<\glshex 1D6F7% upper case phi (maths italic)
12139 \string<\glshex 1D6F8% upper case chi (maths italic)
12140 \string<\glshex 1D6F9% upper case psi (maths italic)

```

```

12141 \string<\glshex 1D6FA% upper case omega (maths italic)
12142 }

```

perGreekIIrules Upper case only (doesn't include upright digamma).

```

12143 \newcommand*{\glxtrMathItalicUpperGreekIIrules}{%
12144 \glshex 1D6E2% upper case alpha (maths italic)
12145 \string<\glshex 1D6E3% upper case beta (maths italic)
12146 \string<\glshex 1D6E4% upper case gamma (maths italic)
12147 \string<\glshex 1D6E5% upper case delta (maths italic)
12148 \string<\glshex 1D6E6% upper case epsilon (maths italic)
12149 \string<\glshex 1D6E7% upper case zeta (maths italic)
12150 \string<\glshex 1D6E8% upper case eta (maths italic)
12151 \string<\glshex 1D6E9% upper case theta (maths italic)
12152 \string<\glshex 1D6F3% upper case theta variant (maths italic)
12153 \string<\glshex 1D6EA% upper case iota (maths italic)
12154 \string<\glshex 1D6EB% upper case kappa (maths italic)
12155 \string<\glshex 1D6EC% upper case lambda (maths italic)
12156 \string<\glshex 1D6ED% upper case mu (maths italic)
12157 \string<\glshex 1D6EE% upper case nu (maths italic)
12158 \string<\glshex 1D6EF% upper case xi (maths italic)
12159 \string<\glshex 1D6F0% upper case omicron (maths italic)
12160 \string<\glshex 1D6F1% upper case pi (maths italic)
12161 \string<\glshex 1D6F2% upper case rho (maths italic)
12162 \string<\glshex 1D6F4% upper case sigma (maths italic)
12163 \string<\glshex 1D6F5% upper case tau (maths italic)
12164 \string<\glshex 1D6F6% upper case upsilon (maths italic)
12165 \string<\glshex 1D6F7% upper case phi (maths italic)
12166 \string<\glshex 1D6F8% upper case chi (maths italic)
12167 \string<\glshex 1D6F9% upper case psi (maths italic)
12168 \string<\glshex 1D6FA% upper case omega (maths italic)
12169 }

```

owerGreekIrules Lower case only (includes upright digamma).

```

12170 \newcommand*{\glxtrMathItalicLowerGreekIrules}{%
12171 \glshex 1D6FC% lower case alpha (maths italic)
12172 \string<\glshex 1D6FD% lower case beta (maths italic)
12173 \string<\glshex 1D6FE% lower case gamma (maths italic)
12174 \string<\glshex 1D6FF% lower case delta (maths italic)
12175 \string<\glshex 1D700% lower case epsilon (maths italic)
12176 \string<\glshex 1D716% lower case epsilon variant (maths italic)
12177 \string<\glshex 03DD% lower case digamma
12178 \string<\glshex 1D701% lower case zeta (maths italic)
12179 \string<\glshex 1D702% lower case eta (maths italic)
12180 \string<\glshex 1D703% lower case theta (maths italic)
12181 \string<\glshex 1D717% lower case theta variant (maths italic)
12182 \string<\glshex 1D704% lower case iota (maths italic)
12183 \string<\glshex 1D705% lower case kappa (maths italic)
12184 \string<\glshex 1D718% lower case kappa variant (maths italic)
12185 \string<\glshex 1D706% lower case lambda (maths italic)

```

```

12186 \string<\glshex 1D707% lower case mu (maths italic)
12187 \string<\glshex 1D708% lower case nu (maths italic)
12188 \string<\glshex 1D709% lower case xi (maths italic)
12189 \string<\glshex 1D70A% lower case omicron (maths italic)
12190 \string<\glshex 1D70B% lower case pi (maths italic)
12191 \string=\glshex 1D71B% lower case pi variant (maths italic)
12192 \string<\glshex 1D70C% lower case rho (maths italic)
12193 \string=\glshex 1D71A% lower case rho variant (maths italic)
12194 \string<\glshex 1D70D% lower case final sigma (maths italic)
12195 \string=\glshex 1D70E% lower case sigma (maths italic)
12196 \string<\glshex 1D70F% lower case tau (maths italic)
12197 \string<\glshex 1D710% lower case upsilon (maths italic)
12198 \string<\glshex 1D711% lower case phi (maths italic)
12199 \string=\glshex 1D719% lower case phi variant (maths italic)
12200 \string<\glshex 1D712% lower case chi (maths italic)
12201 \string<\glshex 1D713% lower case psi (maths italic)
12202 \string<\glshex 1D714% lower case omega (maths italic)
12203 }

```

LowerGreeklIrules Lower case only (doesn't includes upright digamma).

```

12204 \newcommand*{\glxtrMathItalicLowerGreeklIrules}{%
12205 \glshex 1D6FC% lower case alpha (maths italic)
12206 \string<\glshex 1D6FD% lower case beta (maths italic)
12207 \string<\glshex 1D6FE% lower case gamma (maths italic)
12208 \string<\glshex 1D6FF% lower case delta (maths italic)
12209 \string<\glshex 1D700% lower case epsilon (maths italic)
12210 \string=\glshex 1D716% lower case epsilon variant (maths italic)
12211 \string<\glshex 1D701% lower case zeta (maths italic)
12212 \string<\glshex 1D702% lower case eta (maths italic)
12213 \string<\glshex 1D703% lower case theta (maths italic)
12214 \string=\glshex 1D717% lower case theta variant (maths italic)
12215 \string<\glshex 1D704% lower case iota (maths italic)
12216 \string<\glshex 1D705% lower case kappa (maths italic)
12217 \string=\glshex 1D718% lower case kappa variant (maths italic)
12218 \string<\glshex 1D706% lower case lambda (maths italic)
12219 \string<\glshex 1D707% lower case mu (maths italic)
12220 \string<\glshex 1D708% lower case nu (maths italic)
12221 \string<\glshex 1D709% lower case xi (maths italic)
12222 \string<\glshex 1D70A% lower case omicron (maths italic)
12223 \string<\glshex 1D70B% lower case pi (maths italic)
12224 \string=\glshex 1D71B% lower case pi variant (maths italic)
12225 \string<\glshex 1D70C% lower case rho (maths italic)
12226 \string=\glshex 1D71A% lower case rho variant (maths italic)
12227 \string<\glshex 1D70D% lower case final sigma (maths italic)
12228 \string=\glshex 1D70E% lower case sigma (maths italic)
12229 \string<\glshex 1D70F% lower case tau (maths italic)
12230 \string<\glshex 1D710% lower case upsilon (maths italic)
12231 \string<\glshex 1D711% lower case phi (maths italic)
12232 \string=\glshex 1D719% lower case phi variant (maths italic)

```

```

12233 \string<\glshex 1D712% lower case chi (maths italic)
12234 \string<\glshex 1D713% lower case psi (maths italic)
12235 \string<\glshex 1D714% lower case omega (maths italic)
12236 }

```

MathGreekIrules Includes both upright and italic with digamma between epsilon and zeta.

```

12237 \newcommand*{\glxtrMathGreekIrules}{%
12238   \glxtrMathItalicAlpha
12239   \string;\glxtrUpAlpha
12240   \string<\glxtrMathItalicBeta
12241   \string;\glxtrUpBeta
12242   \string<\glxtrMathItalicGamma
12243   \string;\glxtrUpGamma
12244   \string<\glxtrMathItalicDelta
12245   \string;\glxtrUpDelta
12246   \string<\glxtrMathItalicEpsilon
12247   \string;\glxtrUpEpsilon
12248   \string<\glxtrUpDigamma
12249   \string<\glxtrMathItalicZeta
12250   \string;\glxtrUpZeta
12251   \string<\glxtrMathItalicEta
12252   \string;\glxtrUpEta
12253   \string<\glxtrMathItalicTheta
12254   \string;\glxtrUpTheta
12255   \string<\glxtrMathItalicIota
12256   \string;\glxtrUpIota
12257   \string<\glxtrMathItalicKappa
12258   \string;\glxtrUpKappa
12259   \string<\glxtrMathItalicLambda
12260   \string;\glxtrUpLambda
12261   \string<\glxtrMathItalicMu
12262   \string;\glxtrUpMu
12263   \string<\glxtrMathItalicNu
12264   \string;\glxtrUpNu
12265   \string<\glxtrMathItalicXi
12266   \string;\glxtrUpXi
12267   \string<\glxtrMathItalicOmicron
12268   \string;\glxtrUpOmicron
12269   \string<\glxtrMathItalicPi
12270   \string;\glxtrUpPi
12271   \string<\glxtrMathItalicRho
12272   \string;\glxtrUpRho
12273   \string<\glxtrMathItalicSigma
12274   \string;\glxtrUpSigma
12275   \string<\glxtrMathItalicTau
12276   \string;\glxtrUpTau
12277   \string<\glxtrMathItalicUpsilon
12278   \string;\glxtrUpUpsilon
12279   \string<\glxtrMathItalicPhi

```

```

12280 \string;\glxtrUpPhi
12281 \string<\glxtrMathItalicChi
12282 \string;\glxtrUpChi
12283 \string<\glxtrMathItalicPsi
12284 \string;\glxtrUpPsi
12285 \string<\glxtrMathItalicOmega
12286 \string;\glxtrUpOmega
12287 }

```

mathGreekIIrules Includes both upright and italic (digamma not included).

```

12288 \newcommand*{\glxtrMathGreekIIrules}{%
12289 \glxtrMathItalicAlpha
12290 \string;\glxtrUpAlpha
12291 \string<\glxtrMathItalicBeta
12292 \string;\glxtrUpBeta
12293 \string<\glxtrMathItalicGamma
12294 \string;\glxtrUpGamma
12295 \string<\glxtrMathItalicDelta
12296 \string;\glxtrUpDelta
12297 \string<\glxtrMathItalicEpsilon
12298 \string;\glxtrUpEpsilon
12299 \string<\glxtrMathItalicZeta
12300 \string;\glxtrUpZeta
12301 \string<\glxtrMathItalicEta
12302 \string;\glxtrUpEta
12303 \string<\glxtrMathItalicTheta
12304 \string;\glxtrUpTheta
12305 \string<\glxtrMathItalicIota
12306 \string;\glxtrUpIota
12307 \string<\glxtrMathItalicKappa
12308 \string;\glxtrUpKappa
12309 \string<\glxtrMathItalicLambda
12310 \string;\glxtrUpLambda
12311 \string<\glxtrMathItalicMu
12312 \string;\glxtrUpMu
12313 \string<\glxtrMathItalicNu
12314 \string;\glxtrUpNu
12315 \string<\glxtrMathItalicXi
12316 \string;\glxtrUpXi
12317 \string<\glxtrMathItalicOmicron
12318 \string;\glxtrUpOmicron
12319 \string<\glxtrMathItalicPi
12320 \string;\glxtrUpPi
12321 \string<\glxtrMathItalicRho
12322 \string;\glxtrUpRho
12323 \string<\glxtrMathItalicSigma
12324 \string;\glxtrUpSigma
12325 \string<\glxtrMathItalicTau
12326 \string;\glxtrUpTau

```



```

12327 \string<\glxtrMathItalicUpsilon
12328 \string;\glxtrUpUpsilon
12329 \string<\glxtrMathItalicPhi
12330 \string;\glxtrUpPhi
12331 \string<\glxtrMathItalicChi
12332 \string;\glxtrUpChi
12333 \string<\glxtrMathItalicPsi
12334 \string;\glxtrUpPsi
12335 \string<\glxtrMathItalicOmega
12336 \string;\glxtrUpOmega
12337 }

```

\glxtrUpAlpha

```

12338 \newcommand*{\glxtrUpAlpha}{%
12339 \glshex 03B1,% lower case alpha
12340 \glshex 0391% upper case alpha
12341 }

```

\glxtrUpBeta

```

12342 \newcommand*{\glxtrUpBeta}{%
12343 \glshex 03B2,% lower case beta
12344 \glshex 0392% upper case beta
12345 }

```

\glxtrUpGamma

```

12346 \newcommand*{\glxtrUpGamma}{%
12347 \glshex 03B3,% lower case gamma
12348 \glshex 0393% upper case gamma
12349 }

```

\glxtrUpDelta

```

12350 \newcommand*{\glxtrUpDelta}{%
12351 \glshex 03B4,% lower case delta
12352 \glshex 0394% upper case delta
12353 }

```

glxtrUpEpsilon

```

12354 \newcommand*{\glxtrUpEpsilon}{%
12355 \glshex 03B5% lower case epsilon
12356 \string=\glshex 03F5,% lower case epsilon variant
12357 \glshex 0395% upper case epsilon
12358 }

```

glxtrUpDigamma

```

12359 \newcommand*{\glxtrUpDigamma}{%
12360 \glshex 03DD,% lower case digamma
12361 \glshex 03DC% upper case digamma
12362 }

```

`\glxtrUpZeta`

```
12363 \newcommand*{\glxtrUpZeta}{%
12364   \glshex 03B6,% lower case zeta
12365   \glshex 0396% upper case zeta
12366 }
```

`\glxtrUpEta`

```
12367 \newcommand*{\glxtrUpEta}{%
12368   \glshex 03B7,% lower case eta
12369   \glshex 0397% upper case eta
12370 }
```

`\glxtrUpTheta`

```
12371 \newcommand*{\glxtrUpTheta}{%
12372   \glshex 03B8% lower case theta
12373   \string=\glshex 03D1,% lower case theta variant
12374   \glshex 0398% upper case theta
12375 }
```

`\glxtrUpIota`

```
12376 \newcommand*{\glxtrUpIota}{%
12377   \glshex 03B9,% lower case iota
12378   \glshex 0399% upper case iota
12379 }
```

`\glxtrUpKappa`

```
12380 \newcommand*{\glxtrUpKappa}{%
12381   \glshex 03BA% lower case kappa
12382   \string=\glshex 03F0,% lower case kappa variant
12383   \glshex 039A% upper case kappa
12384 }
```

`\glxtrUpLambda`

```
12385 \newcommand*{\glxtrUpLambda}{%
12386   \glshex 03BB,% lower lambda
12387   \glshex 039B% upper case lambda
12388 }
```

`\glxtrUpMu`

```
12389 \newcommand*{\glxtrUpMu}{%
12390   \glshex 03BC,% lower case mu
12391   \glshex 039C% upper case mu
12392 }
```

`\glxtrUpNu`

```
12393 \newcommand*{\glxtrUpNu}{%
12394   \glshex 03BD,% lower case nu
12395   \glshex 039D% upper case nu
12396 }
```

\glxtrUpXi

```
12397 \newcommand*{\glxtrUpXi}{%  
12398   \glshex 03BE,% lower case xi  
12399   \glshex 039E% upper case xi  
12400 }
```

glxtrUpOmicron

```
12401 \newcommand*{\glxtrUpOmicron}{%  
12402   \glshex 03BF,% lower case omicron  
12403   \glshex 039F% upper case omicron  
12404 }
```

\glxtrUpPi

```
12405 \newcommand*{\glxtrUpPi}{%  
12406   \glshex 03C0% lower case pi  
12407   \string=\glshex 03D6,% lower case pi variant  
12408   \glshex 03A0% upper case pi  
12409 }
```

\glxtrUpRho

```
12410 \newcommand*{\glxtrUpRho}{%  
12411   \glshex 03C1% lower case rho  
12412   \string=\glshex 03F1,% lower case rho variant  
12413   \glshex 03A1% upper case rho  
12414 }
```

\glxtrUpSigma

```
12415 \newcommand*{\glxtrUpSigma}{%  
12416   \glshex 03C2% lower case sigma  
12417   \string=\glshex 03C3,% lower case sigma  
12418   \glshex 03A3% upper case sigma  
12419 }
```

\glxtrUpTau

```
12420 \newcommand*{\glxtrUpTau}{%  
12421   \glshex 03C4,% lower case tau  
12422   \glshex 03A4% upper case tau  
12423 }
```

glxtrUpUpsilon

```
12424 \newcommand*{\glxtrUpUpsilon}{%  
12425   \glshex 03C5,% lower case upsilon  
12426   \glshex 03A5% upper case upsilon  
12427 }
```

\glxtrUpPhi

```
12428 \newcommand*{\glxtrUpPhi}{%  
12429   \glshex 03C6% lower case phi
```

```

12430 \string=\glshex 03D5,% lower case phi variant
12431 \glshex 03A6% upper case phi
12432 }

```

$\backslash\mathrm{glxtrUpChi}$

```

12433 \newcommand*{\glxtrUpChi}{%
12434 \glshex 03C7,% lower case chi
12435 \glshex 03A7% upper case chi
12436 }

```

$\backslash\mathrm{glxtrUpPsi}$

```

12437 \newcommand*{\glxtrUpPsi}{%
12438 \glshex 03C8,% lower case psi
12439 \glshex 03A8% upper case psi
12440 }

```

$\backslash\mathrm{glxtrUpOmega}$

```

12441 \newcommand*{\glxtrUpOmega}{%
12442 \glshex 03C9,% lower case omega
12443 \glshex 03A9% upper case omega
12444 }

```

$\mathrm{MathItalicAlpha}$

```

12445 \newcommand*{\glxtrMathItalicAlpha}{%
12446 \glshex 1D6FC,% lower case alpha (maths italic)
12447 \glshex 1D6E2% upper case alpha (maths italic)
12448 }

```

$\mathrm{rMathItalicBeta}$

```

12449 \newcommand*{\glxtrMathItalicBeta}{%
12450 \glshex 1D6FD,% lower case beta (maths italic)
12451 \glshex 1D6E3% upper case beta (maths italic)
12452 }

```

$\mathrm{MathItalicGamma}$

```

12453 \newcommand*{\glxtrMathItalicGamma}{%
12454 \glshex 1D6FE,% lower case gamma (maths italic)
12455 \glshex 1D6E4% upper case gamma (maths italic)
12456 }

```

$\mathrm{MathItalicDelta}$

```

12457 \newcommand*{\glxtrMathItalicDelta}{%
12458 \glshex 1D6FF,% lower case delta (maths italic)
12459 \glshex 1D6E5% upper case delta (maths italic)
12460 }

```

$\mathrm{thItalicEpsilon}$

```

12461 \newcommand*{\glxtrMathItalicEpsilon}{%

```

```

12462 \glshex 1D700% lower case epsilon (maths italic)
12463 \string=\glshex 1D716,% lower case epsilon variant (maths italic)
12464 \glshex 1D6E6% upper case epsilon (maths italic)
12465 }

```

rMathItalicZeta

```

12466 \newcommand*{\glxtrMathItalicZeta}{%
12467 \glshex 1D701,% lower case zeta (maths italic)
12468 \glshex 1D6E7% upper case zeta (maths italic)
12469 }

```

trMathItalicEta

```

12470 \newcommand*{\glxtrMathItalicEta}{%
12471 \glshex 1D702,% lower case eta (maths italic)
12472 \glshex 1D6E8% upper case eta (maths italic)
12473 }

```

MathItalicTheta

```

12474 \newcommand*{\glxtrMathItalicTheta}{%
12475 \glshex 1D703% lower case theta (maths italic)
12476 \string=\glshex 1D717,% lower case theta variant (maths italic)
12477 \glshex 1D6E9% upper case theta (maths italic)
12478 \string=\glshex 1D6F3% upper case theta variant (maths italic)
12479 }

```

rMathItalicIota

```

12480 \newcommand*{\glxtrMathItalicIota}{%
12481 \glshex 1D704,% lower case iota (maths italic)
12482 \glshex 1D6EA% upper case iota (maths italic)
12483 }

```

MathItalicKappa

```

12484 \newcommand*{\glxtrMathItalicKappa}{%
12485 \glshex 1D705% lower case kappa (maths italic)
12486 \string=\glshex 1D718,% lower case kappa variant (maths italic)
12487 \glshex 1D6EB% upper case kappa (maths italic)
12488 }

```

athItalicLambda

```

12489 \newcommand*{\glxtrMathItalicLambda}{%
12490 \glshex 1D706,% lower case lambda (maths italic)
12491 \glshex 1D6EC% upper case lambda (maths italic)
12492 }

```

xtrMathItalicMu

```

12493 \newcommand*{\glxtrMathItalicMu}{%
12494 \glshex 1D707,% lower case mu (maths italic)
12495 \glshex 1D6ED% upper case mu (maths italic)
12496 }

```

xtrMathItalicNu

```
12497 \newcommand*{\glxtrMathItalicNu}{%
12498   \glshex 1D708,% lower case nu (maths italic)
12499   \glshex 1D6EE% upper case nu (maths italic)
12500 }
```

xtrMathItalicXi

```
12501 \newcommand*{\glxtrMathItalicXi}{%
12502   \glshex 1D709,% lower case xi (maths italic)
12503   \glshex 1D6EF% upper case xi (maths italic)
12504 }
```

thItalicOmicron

```
12505 \newcommand*{\glxtrMathItalicOmicron}{%
12506   \glshex 1D70A,% lower case omicron (maths italic)
12507   \glshex 1D6F0% upper case omicron (maths italic)
12508 }
```

xtrMathItalicPi

```
12509 \newcommand*{\glxtrMathItalicPi}{%
12510   \glshex 1D70B% lower case pi (maths italic)
12511   \string=\glshex 1D71B,% lower case pi variant (maths italic)
12512   \glshex 1D6F1% upper case pi (maths italic)
12513 }
```

trMathItalicRho

```
12514 \newcommand*{\glxtrMathItalicRho}{%
12515   \glshex 1D70C% lower case rho (maths italic)
12516   \string=\glshex 1D71A,% lower case rho variant (maths italic)
12517   \glshex 1D6F2% upper case rho (maths italic)
12518 }
```

MathItalicSigma

```
12519 \newcommand*{\glxtrMathItalicSigma}{%
12520   \glshex 1D70D% lower case final sigma (maths italic)
12521   \string=\glshex 1D70E,% lower case sigma (maths italic)
12522   \glshex 1D6F4% upper case sigma (maths italic)
12523 }
```

trMathItalicTau

```
12524 \newcommand*{\glxtrMathItalicTau}{%
12525   \glshex 1D70F,% lower case tau (maths italic)
12526   \glshex 1D6F5% upper case tau (maths italic)
12527 }
```

thItalicUpsilon

```
12528 \newcommand*{\glxtrMathItalicUpsilon}{%
12529   \glshex 1D710,% lower case upsilon (maths italic)
```

```
12530 \glshex 1D6F6% upper case upsilon (maths italic)
12531 }
```

trMathItalicPhi

```
12532 \newcommand*{\glsxtrMathItalicPhi}{%
12533 \glshex 1D711% lower case phi (maths italic)
12534 \string=\glshex 1D719,% lower case phi variant (maths italic)
12535 \glshex 1D6F7% upper case phi (maths italic)
12536 }
```

trMathItalicChi

```
12537 \newcommand*{\glsxtrMathItalicChi}{%
12538 \glshex 1D712,% lower case chi (maths italic)
12539 \glshex 1D6F8% upper case chi (maths italic)
12540 }
```

trMathItalicPsi

```
12541 \newcommand*{\glsxtrMathItalicPsi}{%
12542 \glshex 1D713,% lower case psi (maths italic)
12543 \glshex 1D6F9% upper case psi (maths italic)
12544 }
```

MathItalicOmega

```
12545 \newcommand*{\glsxtrMathItalicOmega}{%
12546 \glshex 1D714,% lower case omega (maths italic)
12547 \glshex 1D6FA% upper case omega (maths italic)
12548 }
```

thItalicPartial

```
12549 \newcommand*{\glsxtrMathItalicPartial}{%
12550 \glshex 1D715% partial differential (maths italic)
12551 }
```

MathItalicNabla

```
12552 \newcommand*{\glsxtrMathItalicNabla}{%
12553 \glshex 1D6FB% nabla (maths italic)
12554 }
```

lsxtrdigitrules Digits from the Basic Latin set and subscript and superscript digit rules.

```
12555 \newcommand*{\glsxtrdigitrules}{%
12556 0\string=\glshex 2080\string=\glshex 2070
12557 \string<1\string=\glshex 2081\string=\glshex 00B9
12558 \string<2\string=\glshex 2082\string=\glshex 00B2
12559 \string<3\string=\glshex 2083\string=\glshex 00B3
12560 \string<4\string=\glshex 2084\string=\glshex 2074
12561 \string<5\string=\glshex 2085\string=\glshex 2075
12562 \string<6\string=\glshex 2086\string=\glshex 2076
12563 \string<7\string=\glshex 2087\string=\glshex 2077
```

```

12564 \string<8\string=\glshex 2088\string=\glshex 2078
12565 \string<9\string=\glshex 2089\string=\glshex 2079
12566 }

```

BasicDigitrules Digits from the Basic Latin set.

```

12567 \newcommand*{\glxtrBasicDigitrules}{%
12568 0\string<1\string<2\string<3\string<4%
12569 \string<5\string<6\string<7\string<8\string<9%
12570 }

```

criptDigitrules Subscript digits.

```

12571 \newcommand*{\glxtrSubScriptDigitrules}{%
12572 \glshex 2080% subscript 0
12573 \string<\glshex 2081% subscript 1
12574 \string<\glshex 2082% subscript 2
12575 \string<\glshex 2083% subscript 3
12576 \string<\glshex 2084% subscript 4
12577 \string<\glshex 2085% subscript 5
12578 \string<\glshex 2086% subscript 6
12579 \string<\glshex 2087% subscript 7
12580 \string<\glshex 2088% subscript 8
12581 \string<\glshex 2089% subscript 9
12582 }

```

criptDigitrules Superscript digits.

```

12583 \newcommand*{\glxtrSuperScriptDigitrules}{%
12584 \glshex 2070% superscript 0
12585 \string<\glshex 00B9% superscript 1
12586 \string<\glshex 00B2% superscript 2
12587 \string<\glshex 00B3% superscript 3
12588 \string<\glshex 2074% superscript 4
12589 \string<\glshex 2075% superscript 5
12590 \string<\glshex 2076% superscript 6
12591 \string<\glshex 2077% superscript 7
12592 \string<\glshex 2078% superscript 8
12593 \string<\glshex 2079% superscript 9
12594 }

```

trfractionrules Vulgar fractions.

```

12595 \newcommand*{\glxtrfractionrules}{%
12596 \glshex 215F% fraction numerator one (1/)
12597 \string<\glshex 2189% zero thirds (0/3 = 0)
12598 \string<\glshex 2152% one tenth (1/10 = 0.1)
12599 \string<\glshex 2151% one ninth (1/9 ~ 0.111)
12600 \string<\glshex 215B% one eighth (1/8 = 0.125)
12601 \string<\glshex 2150% one seventh (1/7 ~ 0.143)
12602 \string<\glshex 2159% one sixth (1/6 ~ 0.167)
12603 \string<\glshex 2155% one fifth (1/5 = 0.2)
12604 \string<\glshex 00BC% one quarter (1/4 = 0.25)

```



```

12605 \string<\glshex 2153% one third ( $1/3 \sim 0.333$ )
12606 \string<\glshex 215C% three eighths ( $3/8 = 0.375$ )
12607 \string<\glshex 2156% two fifths ( $2/5 = 0.4$ )
12608 \string<\glshex 00BD% one half ( $1/2 = 0.5$ )
12609 \string<\glshex 2157% three fifths ( $3/5 = 0.6$ )
12610 \string<\glshex 215D% five eighths ( $5/8 = 0.625$ )
12611 \string<\glshex 2154% two thirds ( $2/3 \sim 0.667$ )
12612 \string<\glshex 00BE% three quarters ( $3/4 = 0.75$ )
12613 \string<\glshex 2158% four fifths ( $4/5 = 0.8$ )
12614 \string<\glshex 215A% five sixths ( $5/6 \sim 0.833$ )
12615 \string<\glshex 215E% seven eighths ( $7/8 = 0.875$ )
12616 }

```

sxtrdialecthook Check for scripts associated with the document dialects.

```

12617 \renewcommand{\@glxsxtrdialecthook}{%
12618   \ifundef\CurrentTrackedScript
12619   {%
12620     \TrackLangIfHasDefaultScript{\CurrentTrackedLanguage}%
12621     {%
12622       \edef\CurrentTrackedScript{%
12623         \TrackLangGetDefaultScript\CurrentTrackedLanguage}%
12624     }%
12625   }%
12626 }%
12627 {}%
12628 \ifdef\CurrentTrackedScript
12629 {%
12630   \let\gls@orgTrackLangRequireDialectPrefix\TrackLangRequireDialectPrefix
12631   \def\TrackLangRequireDialectPrefix{glossariesxtr-}%
12632   \let\CurrentTrackedTag\CurrentTrackedScript
12633   \IfFileExists{\TrackLangRequireDialectPrefix\CurrentTrackedTag.ldf}
12634   {\RequireGlossariesExtraLang{\CurrentTrackedTag}}%
12635   {}%
12636   \let\TrackLangRequireDialectPrefix\gls@orgTrackLangRequireDialectPrefix
12637 }%
12638 {}%
12639 }

```

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.

```

12640 \ifdef\glxsxtr@loaddialect
12641 {%
12642   \@ifpackageloaded{tracklang}
12643   {%
12644     \AnyTrackedLanguages
12645     {%
12646       \ForEachTrackedDialect{\this@dialect}{\glsxtr@loaddialect}%
12647     }%

```

12648 {}%
12649 }
12650 {}
12651 }
12652 {}

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:

```
12653 \NeedsTeXFormat{LaTeX2e}
12654 \ProvidesPackage{glossaries-extra-stylemods}[2018/04/25 v1.30 (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`

```
12655 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
12656 \DeclareOption{all}{%
12657   \appto\@glxtr@loadstyles{%
12658     \RequirePackage{glossary-inline}%
12659     \RequirePackage{glossary-list}%
12660     \RequirePackage{glossary-tree}%
12661     \RequirePackage{glossary-mcols}%
12662     \RequirePackage{glossary-long}%
12663     \RequirePackage{glossary-longragged}%
12664     \RequirePackage{glossary-longbooktabs}%
12665     \RequirePackage{glossary-super}%
12666     \RequirePackage{glossary-superragged}%
12667     \RequirePackage{glossary-bookindex}%
12668   }
12669 }

12670 \DeclareOption*{%
12671   \IfFileExists{glossary-\CurrentOption.sty}
12672   {\eappto\@glxtr@loadstyles{%
12673     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
12674   }%
12675   {%
12676     \PackageError{glossaries-extra-styles}%

```

```

12677     {Unknown option ‘\CurrentOption’}{}%
12678   }%
12679 }

```

Process the package options:

```
12680 \ProcessOptions
```

Load the required packages:

```
12681 \@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`.

```
12682 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

`ewglossarystyle`

```

12683 \providecommand{\renewglossarystyle}[2]{%
12684   \ifcsundef{@glsstyle@#1}%
12685   {%
12686     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
12687   }%
12688   {%
12689     \csdef{@glsstyle@#1}{#2}%
12690   }%
12691 }

```

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.

```

12692 \ifdef{@glsstyle@listdotted}
12693 {%
12694   \renewglossarystyle{listdotted}{%
12695     \setglossarystyle{list}%
12696     \renewcommand*{\glossentry}[2]{%
12697       \item[]\makebox[\glslistdottedwidth][l]{%
12698         \glsentryitem{##1}%
12699         \glstarget{##1}{\glossentryname{##1}}%
12700         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
12701         \glossentrydesc{##1}\glspostdescription}%
12702     \renewcommand*{\subglossentry}[3]{%
12703       \item[]\makebox[\glslistdottedwidth][l]{%
12704         \glssubentryitem{##2}%
12705         \glstarget{##2}{\glossentryname{##2}}%
12706         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
12707         \glossentrydesc{##2}\glspostdescription}%
12708   }

```

```
12709 }
12710 {%
```

Assume the style isn't required if it hasn't already been defined.

```
12711 }
```

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.

```
12712 \ifdef{\@glsstyle@list}
12713 {%
```

listprelocation Space before number list for top-level entries.

```
12714 \newcommand{\glslistprelocation}{\glsxtrprelocation}
```

childprelocation Space before number list for child entries.

```
12715 \newcommand{\glslistchildprelocation}{\glslistprelocation}
```

childpostlocation Full stop after number list.

```
12716 \newcommand{\glslistchildpostlocation}{.}
```

Redefine list to use these commands.

```
12717 \renewglossarystyle{list}{%
12718 \renewenvironment{theglossary}%
12719 {\begin{description}}{\end{description}}%
12720 \renewcommand*{\glossaryheader}{}%
12721 \renewcommand*{\glsgroupheading}[1]{}%
12722 \renewcommand*{\glossentry}[2]{%
12723 \item[\glsentryitem{##1}]%
12724 \glstarget{##1}{\glossentryname{##1}}]
12725 \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
12726 \renewcommand*{\subglossentry}[3]{%
12727 \glssubentryitem{##2}%
12728 \glstarget{##2}{\strut}\space
12729 \glossentrydesc{##2}\glspostdescription
12730 \glslistchildprelocation ##3\glslistchildpostlocation}%
12731 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
12732 }
12733 }
12734 }
```

Similarly for altlist. Since it requires list, the new commands should have been defined above.

```
12735 \ifdef{\@glsstyle@altlist}
12736 {%
12737 \renewglossarystyle{altlist}{%
12738 \setglossarystyle{list}%
12739 \renewcommand*{\glossentry}[2]{%
12740 \item[\glsentryitem{##1}]%
```

```

12741      \glstarget{##1}{\glossentryname{##1}}}%
12742      \mbox{}\par\nobreak\@afterheading
12743      \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
12744      \renewcommand{\subglossentry}[3]{%
12745        \par
12746        \glssubentryitem{##2}%
12747        \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
12748        \glslistchildprelocation ##3}%
12749    }
12750 }
12751 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

12752 \ifdef{\@glsstyle@listgroup}
12753 {%
12754   \renewglossarystyle{listgroup}{%
12755     \setglossarystyle{list}%
12756     \renewcommand*\{\glsgroupheading}[1]{%
12757       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}]}%
12758     \mbox{}\par\nobreak\@afterheading
12759   }%
12760 }
12761 }
12762 {}

```

Similarly for listhypergroup.

```

12763 \ifdef{\@glsstyle@listhypergroup}
12764 {%
12765   \renewglossarystyle{listhypergroup}{%
12766     \setglossarystyle{list}%
12767     \renewcommand*\{\glossaryheader}{%
12768       \glslistnavigationitem{\glsnavigation}}}%
12769     \renewcommand*\{\glsgroupheading}[1]{%
12770       \item[\glslistgroupheaderfmt
12771         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}]}%
12772       \mbox{}\par\nobreak\@afterheading
12773     }%
12774   }
12775 }
12776 {}

```

Similarly for altlistgroup.

```

12777 \ifdef{\@glsstyle@altlistgroup}
12778 {%
12779   \renewglossarystyle{altlistgroup}{%
12780     \setglossarystyle{altlist}%
12781     \renewcommand*\{\glsgroupheading}[1]{%
12782       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}]}%
12783     \mbox{}\par\nobreak\@afterheading
12784   }%
12785 }

```

```

12786 }
12787 {}

    Similarly for altlisthypergroup.
12788 \ifdef{\@glsstyle@altlisthypergroup}
12789 {%
12790   \renewglossarystyle{altlisthypergroup}{%
12791     \setglossarystyle{altlist}%
12792     \renewcommand*\{glossaryheader\}{%
12793       \glslistnavigationitem{\glsnavigation}}%
12794     \renewcommand*\{glsgroupheading}[1]{%
12795       \item[\glslistgroupheaderfmt
12796         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
12797       \mbox{}\par\nobreak\@afterheading
12798     }%
12799   }
12800 }
12801 {}

```

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`.

```

12802 \ifcsdef{@glsstyle@long}
12803 {%
12804   \renewglossarystyle{long}{%
12805     \renewenvironment{theglossary}%
12806       {\begin{longtable}{lp{\glsdescwidth}}}%
12807       {\end{longtable}}%
12808     \renewcommand*\{glossaryheader\}{}%
12809     \renewcommand*\{glsgroupheading}[1]{}%
12810     \renewcommand{\glossentry}[2]{%
12811       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12812       \glossentrydesc{##1}\glspostdescription
12813       \glstrprelocation ##2\tabularnewline
12814     }%
12815     \renewcommand{\subglossentry}[3]{%
12816       &
12817       \glssubentryitem{##2}%
12818       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
12819       \glstrprelocation ##3\tabularnewline
12820     }%
12821     \ifglsnogroupskip
12822       \renewcommand*\{glsgroupskip\}{}%
12823     \else
12824       \renewcommand*\{glsgroupskip\}{ & \tabularnewline}%
12825     \fi
12826   }

```

12827 }
 12828 {}

Three column style:

```
12829 \ifcsdef{@glsstyle@long3col}
12830 {%
12831   \renewglossarystyle{long3col}{%
12832     \renewenvironment{theglossary}%
12833       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
12834       {\end{longtable}}}%
12835     \renewcommand*{\glossaryheader}{}%
12836     \renewcommand*{\glsgroupheading}[1]{}%
12837     \renewcommand{\glossentry}[2]{%
12838       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12839       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
12840     }%
12841     \renewcommand{\subglossentry}[3]{%
12842       &
12843       \glssubentryitem{##2}%
12844       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12845       ##3\tabularnewline
12846     }%
```

Conditional needs to be outside of \glsgroupskip otherwise it can cause “Incomplete \iftrue” errors.

```
12847   \ifglsgroupskip
12848     \renewcommand*{\glsgroupskip}{}%
12849   \else
12850     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
12851   \fi
12852 }
12853 }
12854 {}
```

Four column style:

```
12855 \ifcsdef{@glsstyle@long4col}
12856 {%
12857   \renewglossarystyle{long4col}{%
12858     \renewenvironment{theglossary}%
12859       {\begin{longtable}{llll}}}%
12860       {\end{longtable}}}%
12861     \renewcommand*{\glossaryheader}{}%
12862     \renewcommand*{\glsgroupheading}[1]{}%
12863     \renewcommand{\glossentry}[2]{%
12864       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12865       \glossentrydesc{##1}\glspostdescription &
12866       \glossentrysymbol{##1} &
12867       ##2\tabularnewline
12868     }%
12869     \renewcommand{\subglossentry}[3]{%
12870       &
```



```

12871     \glssubentryitem{##2}%
12872     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12873     \glossentrysymbol{##2} & ##3\tabularnewline
12874 }%

12875 \ifglsgroupskip
12876   \renewcommand*{\glsgroupskip}{}%
12877 \else
12878   \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
12879 \fi
12880 }
12881 }
12882 {}

```

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 `\glxstrprelocation`.

```

12883 \ifcsdef{@glstyle@longragged}
12884 {%
12885   \renewglossarystyle{longragged}{%
12886     \renewenvironment{theglossary}%
12887       {\begin{longtable}[l>{\raggedright}p{\glsdescwidth}]}%
12888       {\end{longtable}}%
12889     \renewcommand*{\glossaryheader}{}%
12890     \renewcommand*{\glsgroupheading}[1]{}%
12891     \renewcommand{\glossentry}[2]{%
12892       \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12893       \glossentrydesc{##1}\glspostdescription\glxstrprelocation ##2%
12894       \tabularnewline
12895     }%
12896     \renewcommand{\subglossentry}[3]{%
12897       &
12898       \glssubentryitem{##2}%
12899       \glstarget{##2}{\strut}\glossentrydesc{##2}%
12900       \glspostdescription\glxstrprelocation ##3%
12901       \tabularnewline
12902     }%
12903     \ifglsgroupskip
12904       \renewcommand*{\glsgroupskip}{}%
12905     \else
12906       \renewcommand*{\glsgroupskip}{& \tabularnewline}%
12907     \fi
12908   }
12909 }

```

12910 {}

Three and four column styles don't use \glxstrprelocation since the number list is in its own column.

```
12911 \ifcsdef{@glsstyle@longragged3col}
12912 {%
12913   \renewglossarystyle{longragged3col}{%
12914     \renewenvironment{theglossary}%
12915       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
12916         >{\raggedright}p{\glspagelistwidth}}}%
12917       {\end{longtable}}}%
12918     \renewcommand*{\glossaryheader}{}%
12919     \renewcommand*{\glsgroupheading}[1]{}%
12920     \renewcommand{\glossentry}[2]{%
12921       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12922       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
12923     }%
12924     \renewcommand{\subglossentry}[3]{%
12925       &
12926       \glssubentryitem{##2}%
12927       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12928       ##3\tabularnewline
12929     }%
12930     \ifglsgroupskip
12931       \renewcommand*{\glsgroupskip}{}%
12932     \else
12933       \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
12934     \fi
12935   }
12936 }
12937 {}
```

Four column style:

```
12938 \ifcsdef{@glsstyle@altlongragged4col}
12939 {%
12940   \renewglossarystyle{altlongragged4col}{%
12941     \renewenvironment{theglossary}%
12942       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
12943         >{\raggedright}p{\glspagelistwidth}}}%
12944       {\end{longtable}}}%
12945     \renewcommand*{\glossaryheader}{}%
12946     \renewcommand*{\glsgroupheading}[1]{}%
12947     \renewcommand{\glossentry}[2]{%
12948       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12949       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
12950       ##2\tabularnewline
12951     }%
12952     \renewcommand{\subglossentry}[3]{%
12953       &
```

```

12954     \glssubentryitem{##2}%
12955     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12956     \glossentrysymbol{##2} & ##3\tabularnewline
12957 }%

12958 \ifglsnogroupskip
12959     \renewcommand*{\glsgroupskip}{}%
12960 \else
12961     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
12962 \fi
12963 }
12964 }
12965 {}

```

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`.

```

12966 \ifcsdef{@glstyle@super}
12967 {%
12968     \renewglossarystyle{super}{%
12969         \renewenvironment{theglossary}%
12970             {\tablehead{}\tabletail}%
12971             \begin{supertabular}{lp{\glsdescwidth}}%
12972             {\end{supertabular}}%
12973         \renewcommand*{\glossaryheader}{}%
12974         \renewcommand*{\glsgroupheading}[1]{}%
12975         \renewcommand{\glossentry}[2]{%
12976             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12977             \glossentrydesc{##1}\glspostdescription
12978             \glxtrprelocation ##2\tabularnewline
12979         }%
12980         \renewcommand{\subglossentry}[3]{%
12981             &
12982             \glssubentryitem{##2}%
12983             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
12984             \glxtrprelocation ##3\tabularnewline
12985         }%
12986         \ifglsnogroupskip
12987             \renewcommand*{\glsgroupskip}{}%
12988         \else
12989             \renewcommand*{\glsgroupskip}{& \tabularnewline}%
12990         \fi
12991     }
12992 }
12993 {}

```

Three column style:

```

12994 \ifcsdef{@glstyle@super3col}

```

```

12995 {%
12996   \renewglossarystyle{super3col}{%
12997     \renewenvironment{theglossary}%
12998       {\tablehead{}\tabletail{}}%
12999       \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
13000     {\end{supertabular}}}%
13001   \renewcommand*{\glossaryheader}{}%
13002   \renewcommand*{\glsgroupheading}[1]{}%
13003   \renewcommand{\glossentry}[2]{%
13004     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
13005     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
13006   }%
13007   \renewcommand{\subglossentry}[3]{%
13008     &
13009     \glssubentryitem{##2}%
13010     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
13011     ##3\tabularnewline
13012   }%

13013   \ifglsgnogroupskip
13014     \renewcommand*{\glsgroupskip}{}%
13015   \else
13016     \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
13017   \fi
13018 }
13019 }
13020 {}

```

Four column styles:

```

13021 \ifcsdef{@glstyle@super4col}
13022 {%
13023   \renewglossarystyle{super4col}{%
13024     \renewenvironment{theglossary}%
13025       {\tablehead{}\tabletail{}}%
13026       \begin{supertabular}{llll}}{%
13027     \end{supertabular}}}%
13028   \renewcommand*{\glossaryheader}{}%
13029   \renewcommand*{\glsgroupheading}[1]{}%
13030   \renewcommand{\glossentry}[2]{%
13031     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
13032     \glossentrydesc{##1}\glspostdescription &
13033     \glossentrysymbol{##1} & ##2\tabularnewline
13034   }%
13035   \renewcommand{\subglossentry}[3]{%
13036     &
13037     \glssubentryitem{##2}%
13038     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
13039     \glossentrysymbol{##2} & ##3\tabularnewline
13040   }%

```

```

13041 \ifglsgroupskip
13042 \renewcommand*{\glsgroupskip}{}%
13043 \else
13044 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
13045 \fi
13046 }
13047 }
13048 {}

```

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`.

```

13049 \ifcsdef{@glstyle@superragged}
13050 {%
13051 \renewglossarystyle{superragged}{%
13052 \renewenvironment{theglossary}%
13053 {\tablehead{}\tabletail{}}%
13054 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}%
13055 {\end{supertabular}}%
13056 \renewcommand*{\glossaryheader}{}%
13057 \renewcommand*{\glsgroupheading}[1]{}%
13058 \renewcommand{\glossentry}[2]{%
13059 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
13060 \glossentrydesc{##1}\glspostdescription\glstrprelocation ##2%
13061 \tabularnewline
13062 }%
13063 \renewcommand{\subglossentry}[3]{%
13064 &
13065 \glssubentryitem{##2}%
13066 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
13067 \glstrprelocation ##3%
13068 \tabularnewline
13069 }%
13070 \ifglsgroupskip
13071 \renewcommand*{\glsgroupskip}{}%
13072 \else
13073 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
13074 \fi
13075 }
13076 }
13077 {}

```

Three column style:

```

13078 \ifcsdef{@glstyle@superragged3col}
13079 {%
13080 \renewglossarystyle{superragged3col}{%

```

```

13081 \renewenvironment{theglossary}%
13082 {\tablehead{}\tabletail{}}%
13083 \begin{supertabular}{1>\raggedright}p{\glsdescwidth}%
13084 >\raggedright}p{\glspagelistwidth}}}%
13085 {\end{supertabular}}%
13086 \renewcommand*\glossaryheader{}%
13087 \renewcommand*\glsgroupheading}[1]{}%
13088 \renewcommand{\glossentry}[2]{%
13089 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
13090 \glossentrydesc{##1}\glspostdescription &
13091 ##2\tabularnewline
13092 }%
13093 \renewcommand{\subglossentry}[3]{%
13094 &
13095 \glssubentryitem{##2}%
13096 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
13097 ##3\tabularnewline
13098 }%
13099 \ifglsgroupskip
13100 \renewcommand*\glsgroupskip{}%
13101 \else
13102 \renewcommand*\glsgroupskip{{ & &\tabularnewline}%
13103 \fi
13104 }
13105 }
13106 {}

```

Four columns:

```

13107 \ifcsdef{@glsstyle@altsuperragged4col}
13108 {%
13109 \renewglossarystyle{altsuperragged4col}{%
13110 \renewenvironment{theglossary}%
13111 {\tablehead{}\tabletail{}}%
13112 \begin{supertabular}{1>\raggedright}p{\glsdescwidth}1%
13113 >\raggedright}p{\glspagelistwidth}}}%
13114 {\end{supertabular}}%
13115 \renewcommand*\glossaryheader{}%
13116 \renewcommand{\glossentry}[2]{%
13117 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
13118 \glossentrydesc{##1}\glspostdescription &
13119 \glossentrysymbol{##1} & ##2\tabularnewline
13120 }%
13121 \renewcommand{\subglossentry}[3]{%
13122 &
13123 \glssubentryitem{##2}%
13124 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
13125 \glossentrysymbol{##2} & ##3\tabularnewline
13126 }%

```

```

13127 \ifglsnogroupskip
13128 \renewcommand*{\glsgroupskip}{}%
13129 \else
13130 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
13131 \fi
13132 }
13133 }
13134 {}

```

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`.

```

13135 \ifdef{\@glsstyle@inline}
13136 {%
13137 \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
13138 \renewcommand*{\glsinlinedescformat}[3]{%
13139 \space#1\glxtrpostdescription}
13140 \renewcommand*{\glsinlinesubdescformat}[3]{%
13141 #1\glxtrpostdescription}

```

Just use `\glxtrpostdescription` instead of `\glspostdescription`.

```

13142 }
13143 {}

```

2.8 Tree Styles

The index style is redefined so that the space before the number list isn't hard coded.

```

13144 \ifdef{\@glsstyle@index}
13145 {

```

`treeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```

13146 \newcommand*{\glstreeprelocation}{\glxtrprelocation}

```

`childprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```

13147 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}

```

```

13148 \renewglossarystyle{index}{%
13149 \renewenvironment{theglossary}%
13150 {\setlength{\parindent}{0pt}%
13151 \setlength{\parskip}{0pt plus 0.3pt}%
13152 \let\item\glstreeitem
13153 \let\subitem\glstreesubitem}

```

```

13154     \let\subsubitem\glstreesubsubitem
13155 }%
13156 {\par}%
13157 \renewcommand*\{\glossaryheader}{}%
13158 \renewcommand*\{\glsgroupheading}[1]{}%
13159 \renewcommand*\{\glossentry}[2]{%
13160     \item\glstentryitem{##1}%
13161     \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
13162     \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
13163     \glstreepredesc \glossentrydesc{##1}\glspostdescription
13164     \glstreeprelocation ##2%
13165 }%
13166 \renewcommand{\subglossentry}[3]{%
13167     \ifcase##1\relax
13168         \item
13169     \or
13170         \subitem
13171         \glssubentryitem{##2}%
13172     \else
13173         \subsubitem
13174     \fi
13175     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
13176     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
13177     \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
13178     \glstreechildprelocation ##3%
13179 }%
13180 \renewcommand*\{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
13181 }
13182 }
13183 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

13184 \ifdef{\@glsstyle@indexgroup}
13185 {%
13186     \renewglossarystyle{indexgroup}{%
13187         \setglossarystyle{index}%
13188         \renewcommand*\{\glsgroupheading}[1]{%
13189             \item\glstreegroupheaderfmt{\glsggetgrouptitle{##1}}%
13190             \nopagebreak\indexspace
13191             \nobreak\@afterheading
13192         }%
13193     }
13194 }
13195 {}

```

Similarly for indexhypergroup.

```

13196 \ifdef{\@glsstyle@indexhypergroup}
13197 {%
13198     \renewglossarystyle{indexhypergroup}{%
13199         \setglossarystyle{index}%

```



```

13200 \renewcommand*{\glossaryheader}{%
13201 \item\glstreenavigationfmt{\glsnavigation}%
13202 \nobreak\@afterheading\indexspace}%
13203 \renewcommand*{\glsgroupheading}[1]{%
13204 \item\glstreegroupheaderfmt
13205 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13206 \nopagebreak\indexspace
13207 \nobreak\@afterheading}%
13208 }%
13209 }
13210 {}

```

Adjust tree style to remove hard coded space before number list.

```

13211 \ifdef{\@glsstyle@tree}
13212 {%
13213 \renewglossarystyle{tree}{%
13214 \renewenvironment{theglossary}%
13215 {\setlength{\parindent}{0pt}%
13216 \setlength{\parskip}{0pt plus 0.3pt}}%
13217 {}%
13218 \renewcommand*{\glossaryheader}{}%
13219 \renewcommand*{\glsgroupheading}[1]{}%
13220 \renewcommand{\glossentry}[2]{%
13221 \hangindent0pt\relax
13222 \parindent0pt\relax
13223 \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
13224 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
13225 \glstreepredesc\glossentrydesc{##1}\glspostdescription
13226 \glstreeprelocation##2\par
13227 }%
13228 \renewcommand{\subglossentry}[3]{%
13229 \hangindent##1\glstreeindent\relax
13230 \parindent##1\glstreeindent\relax
13231 \ifnum##1=1\relax
13232 \glssubentryitem{##2}%
13233 \fi
13234 \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
13235 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
13236 \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
13237 \glstreechildprelocation ##3\par
13238 }%
13239 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
13240 }%
13241 }
13242 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

13243 \ifdef{\@glsstyle@treegroup}
13244 {%
13245 \renewglossarystyle{treegroup}{%

```

```

13246 \setglossarystyle{tree}%
13247 \renewcommand{\glsgroupheading}[1]{\par
13248 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}\par
13249 \nopagebreak\indexspace\nobreak\@afterheading}%
13250 }
13251 }
13252 {}

```

Similarly for treehypergroup

```

13253 \ifdef{\@glsstyle@treehypergroup}
13254 {%
13255 \renewglossarystyle{treehypergroup}{%
13256 \setglossarystyle{tree}%
13257 \renewcommand*\{\glossaryheader}{%
13258 \par\noindent\glstreenavigationfmt{\glsnavigation}\par
13259 \nobreak\@afterheading\indexspace}%
13260 \renewcommand*\{\glsgroupheading}[1]{%
13261 \par\noindent
13262 \glstreegroupheaderfmt
13263 {\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}\par
13264 \nopagebreak\indexspace\nobreak\@afterheading}%
13265 }
13266 }
13267 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

13268 \ifdef{\@glsstyle@treenoname}
13269 {%
13270 \renewglossarystyle{treenoname}{%
13271 \renewenvironment{theglossary}%
13272 {\setlength{\parindent}{0pt}%
13273 \setlength{\parskip}{0pt plus 0.3pt}}%
13274 {}%
13275 \renewcommand*\{\glossaryheader}{}%
13276 \renewcommand*\{\glsgroupheading}[1]{}%
13277 \renewcommand{\glossentry}[2]{%
13278 \hangindent0pt\relax
13279 \parindent0pt\relax
13280 \glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
13281 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
13282 \glstreepredesc\glossentrydesc{##1}\glspostdescription
13283 \glstreeprelocation##2\par
13284 }%
13285 \renewcommand{\subglossentry}[3]{%
13286 \hangindent##1\glstreeindent\relax
13287 \parindent##1\glstreeindent\relax
13288 \ifnum##1=1\relax
13289 \glssubentryitem{##2}%
13290 \fi
13291 \glstarget{##2}{\strut}%

```

```

13292     \glossentrydesc{##2}\glspostdescription\glstreechildprelocation##3\par
13293 }%
13294 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
13295 }
13296 }
13297 {}

```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```

13298 \ifdef{\@glsstyle@treenonamegroup}
13299 {%
13300 \renewglossarystyle{treenonamegroup}{%
13301 \setglossarystyle{treenoname}%
13302 \renewcommand{\glsgroupheading}[1]{\par
13303 \noindent\glstreegroupheaderfmt
13304 {\glsgrouptitle{##1}}}%
13305 \nopagebreak\indexspace\nobreak\@afterheading
13306 }%
13307 }
13308 }
13309 {}

```

Similarly for `treenonamehypergroup`

```

13310 \ifdef{\@glsstyle@treenonamehypergroup}
13311 {%
13312 \renewglossarystyle{treenonamehypergroup}{%
13313 \setglossarystyle{treenoname}%
13314 \renewcommand*{\glossaryheader}{%
13315 \par\noindent\glstreenavigationfmt{\glsnavigation}\par
13316 \nobreak\@afterheading\indexspace}%
13317 \renewcommand*{\glsgroupheading}[1]{%
13318 \par\noindent
13319 \glstreegroupheaderfmt
13320 {\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}%
13321 \nopagebreak\indexspace\nobreak\@afterheading}%
13322 }
13323 }
13324 {}

```

The `almtree` style is redefined to make it easier to make minor adjustments.

```

13325 \ifdef{\@glsstyle@almtree}
13326 {%

```

Only redefine this style if it's already been defined.

symbolDescLocation

```
\glxtralttreeSymbolDescLocation{\langle label \rangle}{\langle location list \rangle}
```

Layout the symbol, description and location for top-level entries.

```

13327 \newcommand{\glxtralttreeSymbolDescLocation}[2]{%

```

```

13328   {%
13329       \let\par\glxstrAltTreePar
13330       \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
13331       \glossentrydesc{#1}\glspostdescription\glstreeprelocation #2\par
13332   }%
13333 }

```

trAltTreeIndent Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

13334 \newlength\glxstrAltTreeIndent

```

lsxtrAltTreePar Multi-paragraph descriptions need to keep the hanging indent.

```

13335 \newcommand{\glxstrAltTreePar}{%
13336     @@par
13337     \glxstrAltTreeSetHangIndent
13338     \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
13339 }

```

symbolDescLocation `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

13340 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
13341     \glxstralttreeSymbolDescLocation{#2}{#3}%
13342 }

```

trreetopindent The original style has to keep computing the width of the name at each entry. This register allows the style to compute it once for the top-level at the start of the glossary.

```

13343 \newlength\glxstrtreetopindent

```

lxstralttreeInit User-level initialisation for the alttree style.

```

13344 \newcommand*{\glxstralttreeInit}{%
13345     \settowidth{\glxstrtreetopindent}{\glstreenamefmt{\glsgetwidestname\space}}%
13346     \glxstrAltTreeIndent=\parindent
13347 }

```

\glsglsetwidest The original `\glsglsetwidest` only uses `\def`. This uses `\gdef`.

```

13348 \newcommand*{\glsglsetwidest}[2][0]{%
13349     \csgdef{@glswidestname\romannumeral#1}{#2}%
13350 }

```

\eglssetwidest The original `\eglssetwidest` only uses `\def`. This uses `\protected@csedef`.

```

13351 \newcommand*{\eglssetwidest}[2][0]{%
13352     \protected@csedef{@glswidestname\romannumeral#1}{#2}%
13353 }

```

`\xglsetwidest` Like the above but uses `\protected@csxdef`.

```
13354 \newcommand*\xglsetwidest}[2][0]{%
13355   \protected@csxdef{#1}{#2}%
13356 }
```

`glsupdatewidest` Only sets if new value is wider than old value.

```
13357 \newcommand*\glsupdatewidest}[2][0]{%
13358   \ifcsundef{#1}%
13359   {\csdef{#1}{#2}}%
13360   {%
13361     \settowidth{#1}{\csuse{#1}}%
13362     \settowidth{#2}%
13363     \ifdim#2>#1%
13364     \csdef{#1}{#2}%
13365   \fi
13366   }%
13367 }
```

`glsupdatewidest` As above but global definition.

```
13368 \newcommand*\gglsupdatewidest}[2][0]{%
13369   \ifcsundef{#1}%
13370   {\csgdef{#1}{#2}}%
13371   {%
13372     \settowidth{#1}{\csuse{#1}}%
13373     \settowidth{#2}%
13374     \ifdim#2>#1%
13375     \csgdef{#1}{#2}%
13376   \fi
13377   }%
13378 }
```

`glsupdatewidest` As `\glsupdatewidest` but expands value.

```
13379 \newcommand*\eglsupdatewidest}[2][0]{%
13380   \ifcsundef{#1}%
13381   {\protected@csedef{#1}{#2}}%
13382   {%
13383     \settowidth{#1}{\csuse{#1}}%
13384     \settowidth{#2}%
13385     \ifdim#2>#1%
13386     \protected@csedef{#1}{#2}%
13387   \fi
13388   }%
13389 }
```

`glsupdatewidest` As above but global.

```
13390 \newcommand*\xglsupdatewidest}[2][0]{%
13391   \ifcsundef{#1}%
13392   {\protected@csxdef{#1}{#2}}%
13393   {%
```

```

13394     \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
13395     \settowidth{\dimen@ii}{#2}%
13396     \ifdim\dimen@ii>\dimen@
13397         \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
13398     \fi
13399 }%
13400 }

```

glswidestname Provide a user-level macro to obtain the widest top-level name.

```

13401 \newcommand*{\glswidestname}{\@glswidestname}

```

glswidestsubname Provide a user-level macro to obtain the widest sub-entry name.

```

13402 \newcommand*{\glswidestsubname}[1]{%
13403     \ifcsundef{@glswidestname\romannumeral#1}%
13404     {\@glswidestname}%
13405     {\csuse{@glswidestname\romannumeral#1}}%
13406 }

```

glswidestTopLevelName CamelCase is easier for long command names. Provide a CamelCase synonym of `\glswidestTopLevelName`.

```

13407 \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.

```

13408 \newrobustcmd*{\glswidestUsedTopLevelName}[1][\@glo@types]{%
13409     \dimen@=0pt\relax
13410     \gls@tmplen=0pt\relax
13411     \foralllglossaries[#1]{\@gls@type}%
13412     {%
13413         \forlgsentries[\@gls@type]{\@glo@label}%
13414         {%
13415             \ifglswidest{\@glo@label}%
13416             {%
13417                 \ifglshasparent{\@glo@label}%
13418                 {}%
13419             }%
13420             \settowidth{\dimen@}{%
13421                 {\glstreenamfmt{\glstentryname{\@glo@label}}}%
13422                 \ifdim\dimen@>\gls@tmplen
13423                     \gls@tmplen=\dimen@
13424                 \eglswidest{\glstentryname{\@glo@label}}%
13425             }%
13426             }%
13427         }%
13428     }%
13429 }%
13430 }%
13431 }

```

destUsedAnyName Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

13432 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
13433 \dimen@=0pt\relax
13434 \gls@tmplen=0pt\relax
13435 \forallglossaries[#1]{\@gls@type}%
13436 {%
13437 \forallgsentries[\@gls@type]{\@glo@label}%
13438 {%
13439 \ifglsused{\@glo@label}%
13440 {%
13441 \settowidth{\dimen@}%
13442 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13443 \ifdim\dimen@>\gls@tmplen
13444 \gls@tmplen=\dimen@
13445 \eglssetwidest{\glsentryname{\@glo@label}}%
13446 \fi
13447 }%
13448 }%
13449 }%
13450 }%
13451 }

```

ndWidestAnyName Like the above but doesn't check is the entry has been used.

```

13452 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
13453 \dimen@=0pt\relax
13454 \gls@tmplen=0pt\relax
13455 \forallglossaries[#1]{\@gls@type}%
13456 {%
13457 \forallgsentries[\@gls@type]{\@glo@label}%
13458 {%
13459 \settowidth{\dimen@}%
13460 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13461 \ifdim\dimen@>\gls@tmplen
13462 \gls@tmplen=\dimen@
13463 \eglssetwidest{\glsentryname{\@glo@label}}%
13464 \fi
13465 }%
13466 }%
13467 }

```

estUsedLevelTwo This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

13468 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
13469 \dimen@=0pt\relax
13470 \dimen@i=0pt\relax
13471 \dimen@ii=0pt\relax
13472 \forallglossaries[#1]{\@gls@type}%
13473 {%

```

```

13474 \forglsentries[\@gls@type]{\@glo@label}%
13475 {%
13476 \ifglsused{\@glo@label}%
13477 {%
13478 \ifglsashasparent{\@glo@label}%
13479 {%
13480 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}}%
13481 \ifglsashasparent{\@glo@parent}%
13482 {%
13483 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
13484 \ifglsashasparent{\@glo@parent}%
13485 {}%
13486 {%
13487 \settowidth{\gls@tmplen}%
13488 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
13489 \ifdim\gls@tmplen>\dimen@ii
13490 \dimen@ii=\gls@tmplen
13491 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
13492 \fi
13493 }%
13494 }%
13495 {%
13496 \settowidth{\gls@tmplen}%
13497 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
13498 \ifdim\gls@tmplen>\dimen@i
13499 \dimen@i=\gls@tmplen
13500 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
13501 \fi
13502 }%
13503 }%
13504 {%
13505 \settowidth{\gls@tmplen}%
13506 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
13507 \ifdim\gls@tmplen>\dimen@
13508 \dimen@=\gls@tmplen
13509 \eglssetwidest{\glsentryname{\@glo@label}}%
13510 \fi
13511 }%
13512 }%
13513 {}%
13514 }%
13515 }%
13516 }

```

`dWidestLevelTwo` This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

13517 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
13518 \dimen@=0pt\relax
13519 \dimen@i=0pt\relax
13520 \dimen@ii=0pt\relax

```



```

13521 \forall glossaries[#1]{\@gls@type}%
13522 {%
13523   \forall sentries[\@gls@type]{\@glo@label}%
13524   {%
13525     \ifglshasparent{\@glo@label}%
13526     {%
13527       \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@label}@parent}}%
13528       \ifglshasparent{\@glo@parent}%
13529       {%
13530         \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@parent}@parent}}%
13531         \ifglshasparent{\@glo@parent}%
13532         {}%
13533         {%
13534           \settowidth{\gls@tmplen}%
13535             {\glstreenamefmt{\glstentryname{\@glo@label}}}%
13536           \ifdim\gls@tmplen>\dimen@ii
13537             \dimen@ii=\gls@tmplen
13538             \eglssetwidest[2]{\glstentryname{\@glo@label}}%
13539           \fi
13540         }%
13541       }%
13542     {%
13543       \settowidth{\gls@tmplen}%
13544         {\glstreenamefmt{\glstentryname{\@glo@label}}}%
13545       \ifdim\gls@tmplen>\dimen@i
13546         \dimen@i=\gls@tmplen
13547         \eglssetwidest[1]{\glstentryname{\@glo@label}}%
13548       \fi
13549     }%
13550   }%
13551   {%
13552     \settowidth{\gls@tmplen}%
13553       {\glstreenamefmt{\glstentryname{\@glo@label}}}%
13554     \ifdim\gls@tmplen>\dimen@
13555       \dimen@=\gls@tmplen
13556       \eglssetwidest{\glstentryname{\@glo@label}}%
13557     \fi
13558   }%
13559 }%
13560 }%
13561 }

```

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.

```

13562 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
13563   \dimen@=0pt\relax
13564   \gls@tmplen=0pt\relax
13565   #2=0pt\relax
13566   \forall glossaries[#1]{\@gls@type}%

```

```

13567  {%
13568    \forglsentries[\@gls@type]{\@glo@label}%
13569  {%
13570    \ifglsused{\@glo@label}%
13571  {%
13572    \settowidth{\dimen@}%
13573      {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13574    \ifdim\dimen@>\gls@tmplen
13575      \gls@tmplen=\dimen@
13576    \eglssetwidest{\glsentryname{\@glo@label}}%
13577    \fi
13578    \settowidth{\dimen@}%
13579      {\glsentrysymbol{\@glo@label}}%
13580    \ifdim\dimen@>#2\relax
13581      #2=\dimen@
13582    \fi
13583  }%
13584  {%
13585  }%
13586  }%
13587  }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

13588  \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
13589    \dimen@=0pt\relax
13590    \gls@tmplen=0pt\relax
13591    #2=0pt\relax
13592    \forallglossaries[#1]{\@gls@type}%
13593  {%
13594    \forglsentries[\@gls@type]{\@glo@label}%
13595  {%
13596    \settowidth{\dimen@}%
13597      {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13598    \ifdim\dimen@>\gls@tmplen
13599      \gls@tmplen=\dimen@
13600    \eglssetwidest{\glsentryname{\@glo@label}}%
13601    \fi
13602    \settowidth{\dimen@}%
13603      {\glsentrysymbol{\@glo@label}}%
13604    \ifdim\dimen@>#2\relax
13605      #2=\dimen@
13606    \fi
13607  }%
13608  }%
13609  }

```

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.

```

13610 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
13611   \dimen@=0pt\relax
13612   \gls@tmplen=0pt\relax
13613   #2=0pt\relax
13614   #3=0pt\relax
13615   \foralllglossaries[#1]{\@gls@type}%
13616   {%
13617     \forglsentries[\@gls@type]{\@glo@label}%
13618     {%
13619       \ifglsused{\@glo@label}%
13620       {%
13621         \settowidth{\dimen@}%
13622           {\glstreenamefmt{\glsentryname{\@glo@label}}}%
13623         \ifdim\dimen@>\gls@tmplen
13624           \gls@tmplen=\dimen@
13625           \eglssetwidest{\glsentryname{\@glo@label}}%
13626         \fi
13627         \settowidth{\dimen@}%
13628           {\glsentrysymbol{\@glo@label}}%
13629         \ifdim\dimen@>#2\relax
13630           #2=\dimen@
13631         \fi
13632         \settowidth{\dimen@}%
13633           {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
13634         \ifdim\dimen@>#3\relax
13635           #3=\dimen@
13636         \fi
13637       }%
13638     }%
13639   }%
13640 }%
13641 }

```

`eSymbolLocation` Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

13642 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
13643   \dimen@=0pt\relax
13644   \gls@tmplen=0pt\relax
13645   #2=0pt\relax
13646   #3=0pt\relax
13647   \foralllglossaries[#1]{\@gls@type}%
13648   {%
13649     \forglsentries[\@gls@type]{\@glo@label}%
13650     {%
13651       \settowidth{\dimen@}%
13652         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
13653       \ifdim\dimen@>\gls@tmplen
13654         \gls@tmplen=\dimen@
13655         \eglssetwidest{\glsentryname{\@glo@label}}%

```

```

13656      \fi
13657      \settowidth{\dimen@}%
13658      {\glstentrysymbol{\@glo@label}}}%
13659      \ifdim\dimen@>#2\relax
13660      #2=\dimen@
13661      \fi
13662      \settowidth{\dimen@}%
13663      {\GlsXtrFormatLocationList{\glstentrynumberlist{\@glo@label}}}%
13664      \ifdim\dimen@>#3\relax
13665      #3=\dimen@
13666      \fi
13667    }%
13668  }%
13669 }

```

AnyNameLocation Like the `\glstFindWidestUsedAnyNameSymbolLocation` 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.

```

13670 \newrobustcmd*{\glstFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
13671   \dimen@=0pt\relax
13672   \glstmplen=0pt\relax
13673   #2=0pt\relax
13674   \forallglossaries[#1]{\@gls@type}%
13675   {%
13676     \forallglstentries[\@gls@type]{\@glo@label}%
13677     {%
13678       \ifglstused{\@glo@label}%
13679       {%
13680         \settowidth{\dimen@}%
13681         {\glstreenamfmt{\glstentryname{\@glo@label}}}%
13682         \ifdim\dimen@>\glstmplen
13683         \glstmplen=\dimen@
13684         \eglissetwidest{\glstentryname{\@glo@label}}%
13685         \fi
13686         \settowidth{\dimen@}%
13687         {\GlsXtrFormatLocationList{\glstentrynumberlist{\@glo@label}}}%
13688         \ifdim\dimen@>#2\relax
13689         #2=\dimen@
13690         \fi
13691       }%
13692     }%
13693   }%
13694 }%
13695 }

```

AnyNameLocation Like the `\glstFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

13696 \newrobustcmd*{\glstFindWidestAnyNameLocation}[2][\@glo@types]{%
13697   \dimen@=0pt\relax
13698   \glstmplen=0pt\relax

```

```

13699      #2=Opt\relax
13700      \forallglossaries[#1]{\@gls@type}%
13701      {%
13702        \forglsentries[\@gls@type]{\@glo@label}%
13703        {%
13704          \settowidth{\dimen@}%
13705            {\glstreenamefmt{\glsentryname{\@glo@label}}}%
13706          \ifdim\dimen@>\gls@tmplen
13707            \gls@tmplen=\dimen@
13708            \eglssetwidest{\glsentryname{\@glo@label}}%
13709          \fi
13710          \settowidth{\dimen@}%
13711            {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
13712          \ifdim\dimen@>#2\relax
13713            #2=\dimen@
13714          \fi
13715        }%
13716      }%
13717    }

```

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`.

```

13718    \newcommand*{\glsxtrComputeTreeIndent}[1]{%
13719      \glstreeindent=\glsxtrtreetopindent\relax
13720    }

```

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.

```

13721    \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
13722      \ifcsundef{@glswidestname\romannumeral#1}%
13723      {%
13724        \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
13725      }%
13726      {%
13727        \settowidth{#3}{\glstreenamefmt{%
13728          \csname @glswidestname\romannumeral#1\endcsname\space}}%
13729      }%
13730    }

```

treeSetHangIndent Set `\hangindent` for top-level entries:

```

13731    \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set \hangindent for sub-entries:

```
13732 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}
```

Redefine alttree:

```
13733 \renewglossarystyle{alttree}{%
13734   \renewenvironment{theglossary}%
13735     {%
13736       \glxtralttreeInit
13737       \def\@gls@prevlevel{-1}%
13738       \mbox{}\par}%
13739     {\par}%
13740   \renewcommand*{\glossaryheader}{}%
13741   \renewcommand*{\glsgroupheading}[1]{}%
13742   \renewcommand{\glossentry}[2]{%
13743     \ifnum\@gls@prevlevel=0\relax
13744     \else
13745       \glxtrComputeTreeIndent{##1}%
13746     \fi
13747     \parindent\glstreeindent
13748     \glxtrAltTreeSetHangIndent
13749     \makebox[0pt][r]%
13750     {%
13751       \glstreenamebox{\glstreeindent}%
13752       {%
13753         \glssentryitem{##1}%
13754         \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
13755       }%
13756     }%
13757     \glxtralttreeSymbolDescLocation{##1}{##2}%
13758     \def\@gls@prevlevel{0}%
13759   }
13760   \renewcommand{\subglossentry}[3]{%
13761     \ifnum##1=1\relax
13762       \glssubentryitem{##2}%
13763     \fi
13764     \ifnum\@gls@prevlevel=##1\relax
13765     \else
13766       \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
13767       \ifnum\@gls@prevlevel<##1\relax
13768         \setlength\glstreeindent\gls@tmplen
13769         \addtolength\glstreeindent\parindent
13770         \parindent\glstreeindent
13771       \else
13772         \ifnum\@gls@prevlevel=0\relax
13773           \glxtrComputeTreeIndent{##2}%
13774         \else
13775           \glxtrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
13776         \fi
13777         \addtolength\parindent{-\glstreeindent}%

```

```

13778         \setlength\glstreeindent\parindent
13779     \fi
13780 \fi
13781 \glxtrAltTreeSetSubHangIndent{##1}%
13782 \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
13783     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
13784 \glxtralttreeSubSymbolDescLocation{##1}{##2}{##3}%
13785 \def\@gls@prevlevel{##1}%
13786 }%
13787 \renewcommand*\{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
13788 }
13789 }%
13790 {%
13791 }

```

Redefine `almtreegroup` so that it discourages a break after group headings. Can't use `\@afterheading` here as it messes with the first item of the group.

```

13792 \ifdef{\@glsstyle@almtreegroup}
13793 {%
13794     \renewglossarystyle{almtreegroup}{%
13795         \setglossarystyle{almtree}%
13796         \renewcommand{\glsgroupheading}[1]{\par
13797             \def\@gls@prevlevel{-1}%
13798             \hangindentOpt\relax
13799             \parindentOpt\relax
13800             \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%
13801         \nopagebreak\indexspace\nopagebreak
13802     }%
13803 }%
13804 }%
13805 {%
13806 }

```

Similarly for `almtreehypergroup`.

```

13807 \ifdef{\@glsstyle@almtreehypergroup}
13808 {%
13809     \renewglossarystyle{almtreehypergroup}{%
13810         \setglossarystyle{almtree}%
13811         \renewcommand*\{\glossaryheader}{%
13812             \par
13813             \def\@gls@prevlevel{-1}%
13814             \hangindentOpt\relax
13815             \parindentOpt\relax
13816             \glstreenavigationfmt{\glsnavigation}\par\indexspace
13817         }%
13818         \renewcommand*\{\glsgroupheading}[1]{%
13819             \par
13820             \def\@gls@prevlevel{-1}%
13821             \hangindentOpt\relax
13822             \parindentOpt\relax

```

```

13823     \glstreegroupheaderfmt
13824     {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
13825     \nopagebreak\indexspace\nopagebreak
13826   }%
13827 }
13828 }%
13829 {%
13830 }

```

2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

13831 \ifdef{\@glsstyle@mcolindexgroup}
13832 {%
13833   \renewglossarystyle{mcolindexgroup}{%
13834     \setglossarystyle{mcolindex}%
13835     \renewcommand*\{glsgroupheading}[1]{%
13836       \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
13837       \nopagebreak\indexspace\nobreak\@afterheading
13838     }%
13839   }
13840 }%
13841 {%
13842 }

```

Similarly for mcolindexhypergroup.

```

13843 \ifdef{\@glsstyle@mcolindexhypergroup}
13844 {%
13845   \renewglossarystyle{mcolindexhypergroup}{%
13846     \setglossarystyle{mcolindex}%
13847     \renewcommand*\{glossaryheader}{%
13848       \item\glstreenavigationfmt{\glsnavigation}%
13849       \indexspace
13850     }%
13851     \renewcommand*\{glsgroupheading}[1]{%
13852       \item\glstreegroupheaderfmt
13853         {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
13854       \nopagebreak\indexspace\nobreak\@afterheading
13855     }%
13856   }
13857 }%
13858 {%
13859 }

```

Similarly for mcolindexspannav.

```

13860 \ifdef{\@glsstyle@mcolindexspannav}
13861 {%
13862   \renewglossarystyle{mcolindexspannav}{%
13863     \setglossarystyle{index}%

```



```

13864 \renewenvironment{theglossary}%
13865 {%
13866 \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]]%
13867 \setlength{\parindent}{0pt}%
13868 \setlength{\parskip}{0pt plus 0.3pt}%
13869 \let\item\glstreeitem}%
13870 {\end{multicols}}}%
13871 \renewcommand*{\glsgroupheading}[1]{%
13872 \item\glstreegroupheaderfmt
13873 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13874 \nopagebreak\indexspace\nobreak\@afterheading
13875 }%
13876 }
13877 }%
13878 {%
13879 }

```

Similarly for mcoltreegroup.

```

13880 \ifdef{\@glstyle@mcoltreegroup}
13881 {%
13882 \renewglossarystyle{mcoltreegroup}{%
13883 \setglossarystyle{mcoltree}%
13884 \renewcommand{\glsgroupheading}[1]{\par
13885 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
13886 \nopagebreak\indexspace\nobreak\@afterheading
13887 }%
13888 }
13889 }%
13890 {%
13891 }

```

Similarly for mcoltreehypergroup.

```

13892 \ifdef{\@glstyle@mcoltreehypergroup}
13893 {%
13894 \renewglossarystyle{mcoltreehypergroup}{%
13895 \setglossarystyle{mcoltree}%
13896 \renewcommand*{\glossaryheader}{%
13897 \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace
13898 }%
13899 \renewcommand*{\glsgroupheading}[1]{%
13900 \par\noindent
13901 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13902 \nopagebreak\indexspace\nobreak\@afterheading
13903 }%
13904 }
13905 }%
13906 {%
13907 }

```

Similarly for mcoltreespannav.

```

13908 \ifdef{\@glstyle@mcoltreespannav}

```

```

13909 {%
13910   \renewglossarystyle{mcoltreespannav}{%
13911     \setglossarystyle{tree}%
13912     \renewenvironment{theglossary}%
13913     {%
13914       \begin{multicols}{\glsmcols}%
13915       [\noindent\glstreenavigationfmt{\glsnavigation}]%
13916       \setlength{\parindent}{0pt}%
13917       \setlength{\parskip}{0pt plus 0.3pt}%
13918     }%
13919     {\end{multicols}}}%
13920   \renewcommand*{\glsgroupheading}[1]{%
13921     \par\noindent
13922     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13923     \nopagebreak\indexspace\nobreak\@afterheading
13924   }%
13925 }
13926 }%
13927 {%
13928 }

```

Similarly for mcoltreenonamegroup.

```

13929 \ifdef{\@glsstyle@mcoltreenonamegroup}
13930 {%
13931   \renewglossarystyle{mcoltreenonamegroup}{%
13932     \setglossarystyle{mcoltreenoname}%
13933     \renewcommand*{\glsgroupheading}[1]{\par
13934       \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
13935     \nopagebreak\indexspace\nobreak\@afterheading
13936   }%
13937 }
13938 }%
13939 {%
13940 }

```

Similarly for mcoltreenonamehypergroup.

```

13941 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
13942 {%
13943   \renewglossarystyle{mcoltreenonamehypergroup}{%
13944     \setglossarystyle{mcoltreenoname}%
13945     \renewcommand*{\glossaryheader}{%
13946       \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace}%
13947     \renewcommand*{\glsgroupheading}[1]{%
13948       \par\noindent
13949       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13950       \nopagebreak\indexspace\nobreak\@afterheading}%
13951   }
13952 }%
13953 {%
13954 }

```

Similarly for mcoltreenonamespannav.

```

13955 \ifdef{\@glsstyle@mcoltreenonamespannav}
13956 {%
13957   \renewglossarystyle{mcoltreenonamespannav}{%
13958     \setglossarystyle{treenoname}%
13959     \renewenvironment{theglossary}%
13960     {%
13961       \begin{multicols}{\glsmcols}%
13962       [\noindent\glstreenavigationfmt{\glsnavigation}}%
13963       \setlength{\parindent}{0pt}%
13964       \setlength{\parskip}{0pt plus 0.3pt}%
13965     }%
13966     {\end{multicols}}}%
13967   \renewcommand*{\glsgroupheading}[1]{%
13968     \par\noindent
13969     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
13970     \nopagebreak\indexspace\nobreak\@afterheading}%
13971   }
13972 }%
13973 {%
13974 }

```

mcolalmtree needs adjusting so that it uses \glxtralttreeInit This doesn't use \mbox{} \par which would unbalance the top of the columns.

```

13975 \ifdef{\@glsstyle@mcolalmtree}
13976 {%
13977   \renewglossarystyle{mcolalmtree}{%
13978     \setglossarystyle{almtree}%
13979     \renewenvironment{theglossary}%
13980     {%
13981       \glxtralttreeInit
13982       \def\@gls@prevlevel{-1}%
13983       \begin{multicols}{\glsmcols}%
13984     }%
13985     {\par\end{multicols}}}%
13986   }
13987 }%
13988 {%
13989 }

```

Redefine mcolalmtreegroup to discourage page breaks after the group headings.

```

13990 \ifdef{\@glsstyle@mcolalmtreegroup}
13991 {%
13992   \renewglossarystyle{mcolalmtreegroup}{%
13993     \setglossarystyle{mcolalmtree}%
13994     \renewcommand{\glsgroupheading}[1]{\par
13995       \def\@gls@prevlevel{-1}%
13996       \hangindent0pt\relax
13997       \parindent0pt\relax
13998       \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%

```

```

13999      \nopagebreak\indexspace\nopagebreak
14000  }%
14001  }
14002 }%
14003 {%
14004 }

```

Similarly for mcolalmtreehypergroup.

```

14005 \ifdef{\@glsstyle@mcolalmtreehypergroup}
14006 {%
14007   \renewglossarystyle{mcolalmtreehypergroup}{%
14008     \setglossarystyle{mcolalmtree}%
14009     \renewcommand*{\glossaryheader}{%
14010       \par
14011       \def\@gls@prevlevel{-1}%
14012       \hangindent0pt\relax
14013       \parindent0pt\relax
14014       \glstreenavigationfmt{\glsnavigation}%
14015       \par\indexspace
14016     }%
14017     \renewcommand*{\glsgroupheading}[1]{%
14018       \par
14019       \def\@gls@prevlevel{-1}%
14020       \hangindent0pt\relax
14021       \parindent0pt\relax
14022       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
14023       \nopagebreak\indexspace\nopagebreak
14024     }%
14025   }
14026 }%
14027 {%
14028 }

```

Similarly for mcolalmtreespannav.

```

14029 \ifdef{\@glsstyle@mcolalmtreespannav}
14030 {%
14031   \renewglossarystyle{mcolalmtreespannav}{%
14032     \setglossarystyle{almtree}%
14033     \renewenvironment{theglossary}%
14034     {%
14035       \glsxtralmtreeInit
14036       \def\@gls@prevlevel{-1}%
14037       \begin{multicols}{\glsncols}%
14038         [\noindent\glstreenavigationfmt{\glsnavigation}]%
14039     }%
14040     {\par\end{multicols}}%
14041     \renewcommand*{\glsgroupheading}[1]{%
14042       \par
14043       \def\@gls@prevlevel{-1}%
14044       \hangindent0pt\relax

```

```

14045      \parindent0pt\relax
14046      \glstreegroupheaderfmt{\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
14047      \nopagebreak\indexspace\nopagebreak
14048    }%
14049  }
14050}%
14051{%
14052}

      Reset the default style
14053\ifx\@glossary@default@style\relax
14054\else
14055  \setglossarystyle{\@glxtr@current@style}
14056\fi

```

3 bookindex style (glossary-bookindex.sty)

3.1 Package Initialisation and Options

```
14057 \NeedsTeXFormat{LaTeX2e}
14058 \ProvidesPackage{glossary-bookindex}[2018/04/25 v1.30 (NLCT)]
```

Load required packages.

```
14059 \RequirePackage{multicol}
14060 \RequirePackage{glossary-tree}
```

`trbookindexcols` Number of columns.

```
14061 \newcommand{\glstrbookindexcols}{2}
```

`trbookindexname` Format used for top-level entries. (Argument is the label.)

```
14062 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

`bookindexsubname` Format used for sub entries.

```
14063 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

`glstrprelocation` Provide in case glossaries-stylemods isn't loaded.

```
14064 \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.

```
14065 \newcommand*{\glstrbookindexprelocation}[1]{%
14066   \glstrifhasfield{location}{#1}%
14067   {,\glstrprelocation}%
14068   {\glstrprelocation}%
14069 }
```

`glstrsubprelocation` Separator used before location list for sub-entries.

```
14070 \newcommand*{\glstrbookindexsubprelocation}[1]{%
14071   \glstrbookindexprelocation{#1}%
14072 }
```

`glstrparentchildsep` Separator used between top-level parent and child entry.

```
14073 \newcommand{\glstrbookindexparentchildsep}{\nopagebreak}
```

`glstrparentsubchildsep` Separator used between sub-level parent and child entry.

```
14074 \newcommand{\glstrbookindexparentsubchildsep}{\glstrbookindexparentchildsep}
```

`bookindexbetween` Between two top-level entries identified by the labels in the arguments.

```
14075 \newcommand{\glxstrbookindexbetween}[2]{}

indexsubbetween Between two level 1 entries identified by the labels in the arguments.
14076 \newcommand{\glxstrbookindexsubbetween}[2]{}

exsubsubbetween Between two level 2 entries identified by the labels in the arguments.
14077 \newcommand{\glxstrbookindexsubsubbetween}[2]{}

indexatendgroup At the end of a letter group. The argument is the index of the last top-level entry.
14078 \newcommand{\glxstrbookindexatendgroup}[1]{}

exsubatendgroup At the end of a letter group. The argument is the index of the last level 1 entry.
14079 \newcommand{\glxstrbookindexsubatendgroup}[1]{}

subsubatendgroup At the end of a letter group. The argument is the index of the last level 2 entry.
14080 \newcommand{\glxstrbookindexsubsubatendgroup}[1]{}

kindexgroupskip Group separator.
14081 \newcommand{\glxstrbookindexgroupskip}{\ifglsnogroupskip\else\indexspace\fi}

Format group title.

dexformatheader Group separator.
14082 \newcommand*{\glxstrbookindexformatheader}[1]{%
14083 \par{\centering\glstreegroupheaderfmt{#1}\par}%
14084 }

bookindexbookmark Book mark group heading if supported.
14085 \ifdef\pdfbookmark
14086 {%
14087 \newcommand*{\glxstrbookindexbookmark}[2]{%
14088 \ifdefstring{\@@glossarysec}{chapter}%
14089 {\pdfbookmark[1]{#1}{#2}}%
14090 {\pdfbookmark[2]{#1}{#2}}%
14091 }
14092 }
14093 {%
14094 \newcommand*{\glxstrbookindexbookmark}[2]{}
14095 }

kindexcolspread
14096 \newcommand*{\glxstrbookindexcolspread}{}

dexmulticolseenv
14097 \newcommand*{\glxstrbookindexmulticolseenv}{multicols}
```

Define the style.

```

14098 \newglossarystyle{bookindex}{%
14099   \setglossarystyle{index}%
14100   \renewenvironment{theglossary}%
14101   {%
14102     \ifdefempty{glxstrbookindexcolspread}
14103     {%
14104       \expandafter\begin\expandafter{\glxstrbookindexmulticolseenv}%
14105       {\glxstrbookindexcols}%
14106     }%
14107     {%
14108       \expandafter\begin\expandafter{\glxstrbookindexmulticolseenv}%
14109       {\glxstrbookindexcols}[\glxstrbookindexcolspread]%
14110     }%
14111     \setlength{\parindent}{0pt}%
14112     \setlength{\parskip}{0pt plus 0.3pt}%
14113     \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
14114     \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
14115     \let\@glxstr@bookindex@between\@gobble
14116     \let\@glxstr@bookindex@subbetween\@gobble
14117     \let\@glxstr@bookindex@subsubbetween\@gobble
14118     \let\@glxstr@bookindex@atendgroup\relax
14119     \let\@glxstr@bookindex@subatendgroup\relax
14120     \let\@glxstr@bookindex@subsubatendgroup\relax
14121     \let\@glxstr@bookindex@groupskip\relax
14122   }%
14123   {%

```

Do end group hooks.

```

14124     \@glxstr@bookindex@subsubatendgroup
14125     \@glxstr@bookindex@subatendgroup
14126     \@glxstr@bookindex@atendgroup

```

End multicol environment.

```

14127     \expandafter\end\expandafter{\glxstrbookindexmulticolseenv}%
14128   }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

14129   \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

14130   \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

14131     \@glxstr@bookindex@between{##1}%

```

Update separators.

```

14132     \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
14133     \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
14134     \let\@glxstr@bookindex@subbetween\@gobble
14135     \let\@glxstr@bookindex@subsubbetween\@gobble
14136     \edef\@glxstr@bookindex@between{%

```



```

14137     \noexpand\glxstrbookindexbetween{##1}%
14138 }%
14139 \edef\@glxstr@bookindex@atendgroup{%
14140     \noexpand\glxstrbookindexatendgroup{##1}%
14141 }%
14142 \let\@glxstr@bookindex@subatendgroup\relax
14143 \let\@glxstr@bookindex@subsubatendgroup\relax

```

Format entry.

```

14144 \glstreeitem
14145     \glstryitem{##1}%
14146     \glstarget{##1}{\glxstrbookindexname{##1}}%
14147     \glxstrbookindexprelocation{##1}##2%
14148 }%
14149 \renewcommand{\subglossentry}[3]{%
14150     \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```

14151     \glstreeitem
14152     \or

```

Level 1.

```

14153     \@glxstr@bookindex@sep
14154     \@glxstr@bookindex@subbetween{##2}%
14155     \let\@glxstr@bookindex@sep\relax

```

Update separators.

```

14156     \let\@glxstr@bookindex@subsubbetween\@gobble
14157     \let\@glxstr@bookindex@subsep\glxstrbookindexparentschildsep
14158     \edef\@glxstr@bookindex@subbetween{%
14159         \noexpand\glxstrbookindexsubbetween{##2}%
14160     }%
14161     \edef\@glxstr@bookindex@atsubendgroup{%
14162         \noexpand\glxstrbookindexatsubendgroup{##1}%
14163     }%

```

Start sub-item.

```

14164     \glstreesubitem
14165     \glssubentryitem{##2}%
14166     \else

```

All other levels.

```

14167     \@glxstr@bookindex@subsep
14168     \@glxstr@bookindex@subsubbetween{##2}%

```

Update separators.

```

14169     \let\@glxstr@bookindex@subsep\relax
14170     \edef\@glxstr@bookindex@subsubbetween{%
14171         \noexpand\glxstrbookindexsubsubbetween{##2}%
14172     }%
14173     \edef\@glxstr@bookindex@atsubsubendgroup{%
14174         \noexpand\glxstrbookindexatsubsubendgroup{##1}%
14175     }%

```

Start sub-sub-item.

```
14176 \glstreesubsubitem
14177 \fi
```

Format entry.

```
14178 \glstarget{##2}{\glxtrbookindexsubname{##2}}%
14179 \glxtrbookindexsubprelocation{##2}##3%
14180 }%
```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
14181 \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
14182 \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
14183 \@glxtr@bookindex@subsubatendgroup
14184 \@glxtr@bookindex@subatendgroup
14185 \@glxtr@bookindex@atendgroup
14186 \@glxtr@bookindexgroupskip
```

Update separators.

```
14187 \let\@glxtr@bookindexgroupskip\glxtrbookindexgroupskip
14188 \let\@glxtr@bookindex@between\@gobble
14189 \let\@glxtr@bookindex@atendgroup\relax
14190 \let\@glxtr@bookindex@subatendgroup\relax
14191 \let\@glxtr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
14192 \glxtrgetgrouptitle{##1}{\thisgrptitle}%
```

Do the PDF bookmark if supported.

```
14193 \glxtrbookindexbookmark{\thisgrptitle}{index.##1}%
```

Format the group title.

```
14194 \glxtrbookindexformatheader{\thisgrptitle}%
14195 \nopagebreak\indexspace\nopagebreak\@afterheading
14196 }%
14197 }
```

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.

```
14198 \newcommand{\glxtrbookindexthepage}{%
14199 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
14200 }
```

bookindexmarkentry Writes entry information to the .aux file. The argument is the entry label.

```
14201 \newcommand*{\glxtrbookindexmarkentry}[1]{%
14202   \protected@write\@auxout
14203   {\let\glxtrbookindexthepage\relax}%
14204   {\string\glxtr@setbookindexmark{\glxtrbookindexthepage}{#1}}%
14205 }
```

etbookindexmark

```
14206 \newcommand*{\glxtr@setbookindexmark}[2]{%
14207   \ifcsundef{glxtr@idxfirstmark@#1}%
14208   {\csgdef{glxtr@idxfirstmark@#1}{#2}}%
14209   {}%
14210   \csgdef{glxtr@idxlastmark@#1}{#2}%
14211 }
```

indexfirstmarkfmt

```
14212 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
14213   \glstryname{#1}%
14214 }
```

indexfirstmark

```
14215 \newcommand*{\glxtrbookindexfirstmark}{%
14216   \letcs{glxtr@label}{glxtr@idxfirstmark@glxtrbookindexthepage}%
14217   \ifdefglxtr@label
14218   {\glxtrbookindexfirstmarkfmt{glxtr@label}}%
14219   {}%
14220 }
```

indexlastmarkfmt

```
14221 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
14222   \glstryname{#1}%
14223 }
```

okindexlastmark

```
14224 \newcommand*{\glxtrbookindexlastmark}{%
14225   \letcs{glxtr@label}{glxtr@idxlastmark@glxtrbookindexthepage}%
14226   \ifdefglxtr@label
14227   {\glxtrbookindexlastmarkfmt{glxtr@label}}%
14228   {}%
14229 }
```

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	69
General: Initial experimental release	5	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	69
\Glsfmtshort: new	312	\@Glstext@: added redefinition	64
\glsfmtshort: new	312	\@Glsuseri@: added redefinition	69
\Glsfmtshortpl: new	313	\@Glsuserii@: added redefinition	70
\glsfmtshortpl: new	312	\@Glsuseriii@: added redefinition	70
short: switched inline full form to short		\@Glsuseriv@: added redefinition	70
(long)	215	\@Glsuserv@: added redefinition	71
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	71
\@ACRlong: added redefinition	74	\@acrlong: added redefinition	73
\@ACRlongpl: added redefinition	75	\@acrlongpl: added redefinition	74
\@ACRshort: added redefinition	72	\@acrshort: added redefinition	71
\@ACRshortpl: added redefinition	73	\@acrshortpl: added redefinition	72
\@Acrlong: added redefinition	74	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	75	argument	57
\@Acrshort: added redefinition	72	\@glsdescplural@: added redefinition .	68
\@Acrshortpl: added redefinition	73	\@glsfirst@: added redefinition	65
\@GLSdesc@: added redefinition	68	\@glsfirstplural@: added redefinition	66
\@GLSdescplural@: added redefinition .	68	\@glsplural@: added redefinition	66
\@GLSfirst@: added redefinition	65	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	67	redefinition	69
\@GLSname@: added redefinition	67	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	66	new	126
\@GLSSymbol@: added redefinition	69	\@glsxtr@field@linkdefs: new	64
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	183
redefinition	69	\@print@glossary: added redefinition	123
\@GLStext@: added redefinition	64	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	70	\abbrvdefaultfont	189
\@GLSuserii@: added redefinition	70	\glsaccessdesc: new	151
\@GLSuseriii@: added redefinition	70	\glsaccessdescplural: new	151
\@GLSuseriv@: added redefinition	71	\glsaccessfirst: new	148
\@GLSuserv@: added redefinition	71	\glsaccessfirstplural: new	149
\@GLSuservi@: added redefinition	71	\Glsaccesslong: new	153
\@GLSdesc@: added redefinition	68	\glsaccesslong: new	153
\@GLSdescplural@: added redefinition .	68	\glsaccessname: new	147
\@GLSfirst@: added redefinition	65	\glsaccessplural: new	148
\@GLSfirstplural@: added redefinition	66	\Glsaccessshort: new	152
\@GLSname@: added redefinition	67	\glsaccessshort: new	152
\@GLSplural@: added redefinition	66	\Glsaccessshortpl: new	153

\glsaccesssshortpl: new	153	\@cGLSpl: new	100
\glsaccessssymbol: new	150	\@cGLSpl@: new	100
\glsaccessssymbolplural: new	150	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	147	new	95
\glstentryfmt: added check for short ..	56	\cGLS: new	99
\glslongpltok: new	183	\cGLSformat: new	100
\glsshortpltok: new	183	\cGLSpl: new	100
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	100
name in \setkeys	185	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	16
for plural	180	\glsenableentrycount: new	95
\GLSxtrlongpl: new	199	\glsfirstabbrvdefaultfont: new ..	189
\Glsxtrlongpl: new	199	\glsfirstlongdefaultfont: new ...	189
\glsxtrlongpl: new	198	\Glsfmtfirst: new	315
\glsxtrNoGlossaryWarning: new	21	\glsfmtfirst: new	315
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	316
new	180	\glsfmtfirstpl: new	315
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	314
new	180	\glsfmtplural: new	314
\glsxtrpostlinkendsentence: new ..	180	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	197	\Glsxtrtitleshort	312
\Glsxtrshortpl: new	197	renamed from \glstentryfmtshort ..	312
\glsxtrshortpl: new	196	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort	312
\glslabeltok	210	renamed from \glstentryfmtshort ..	312
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	208	\Glsxtrtitleshortpl	313
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl	313
redefinition of \acronymtype	17	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	312
\Glsxtrshort	312	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl	312
\glsxtrshort	312	\Glsfmttext: new	314
\Glsfmtshortpl: changed to use		\glsfmttext: new	314
\glsxtrshortpl	313	\glshasattribute: new	159
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	158
\glsxtrshortpl	312	\glsxtremsuffix: new	251
\glsxtrifemptyglossary: new	27	\GlsXtrEnableEntryCounting: new ..	94
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new	97
argument	162	\glsxtrscfont: new	223
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new	223
argument	162	\glsxtrsmfont: new	237
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new	237
default type to \acronymtype	108	short-em: new	258
\newterm: fixed name argument	161	short-em-desc: new	260
0.5 (2015-12-07)		short-em-footnote: new	269
\@cGLS: new	99	short-em-long: new	255
\@cGLS@: new	100	short-em-long-desc: new	256

short-em-postfootnote: new	271	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new	233	attribute	303
short-sc-postfootnote: new	235	\Glsxtrheadtext: now uses headuc	
short-sm: new	241	attribute	306
short-sm-desc: new	242	\glxtrheadtext: now uses headuc	
short-sm-footnote: new	248	attribute	305
short-sm-long: new	239	short-em-footnote: switch off regular	
short-sm-long-desc: new	240	attribute if set	269
short-sm-postfootnote: new	249	short-long: switch off regular attribute	
long-noshort-em: new	262	if set	209
long-noshort-em-desc: new	266	short-long-desc: switch off regular	
long-noshort-sm: new	244	attribute if set	210
long-noshort-sm-desc: new	246	short-sc-footnote: switch off regular	
long-short-em: new	252	attribute if set	234
long-short-em-desc: new	253	short-sm-footnote: switch off regular	
long-short-sm: new	238	attribute if set	248
long-short-sm-desc: new	239	long-short: switch off regular attribute	
0.5.1 (2015-12-02)		if set	207
\Glsaccessstext: new	148	long-short-desc: switch off regular	
0.5.1 (2015-12-07)		attribute if set	208
\@glxtr@doaccsupp: new	21	long-short-sc-desc: switch off regular	
General: removed \ifglxtruseuchead	302	attribute if set	225
\Glsaccessdesc: new	151	footnote: switch off regular attribute if	
\Glsaccessdescplural: new	152	set	211
\Glsaccessfirst: new	149	postfootnote: switch off regular	
\Glsaccessfirstplural: new	149	attribute if set	213
\Glsaccessname: new	147	0.5.2 (2015-12-08)	
\Glsaccessplural: new	148	\@GLSdesc@: added accessibility support	68
\Glsaccesssymbol: new	150	\@GLSdescplural@: added accessibility	
\Glsaccesssymbolplural: new	150	support	68
\glxtrheadfirst: now uses headuc		\@GLSfirst@: added accessibility	
attribute	307	support	65
\glxtrheadfirst: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	307	support	67
\Glsxtrheadfirstplural: now uses		\@GLSname@: added accessibility support	67
headuc attribute	308	\@GLSplural@: added accessibility	
\glxtrheadfirstplural: now uses		support	66
headuc attribute	308	\@GLSsymbol@: added accessibility	
\Glsxtrheadplural: now uses headuc		support	69
attribute	306	\@GLSsymbolplural@: added	
\glxtrheadplural: now uses headuc		accessibility support	69
attribute	306	\@GLStext@: added accessibility support	64
\Glsxtrheadshort: now uses headuc		\@GLSdesc@: added accessibility support	68
attribute	303	\@GLSdescplural@: added accessibility	
\glxtrheadshort: now uses headuc		support	68
attribute	303	\@GLSfirst@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support	65
attribute	304	\@GLSfirstplural@: added accessibility	
		support	66

\@Glsname@: add accessibility support ..	67	\GLSaccesssymbolplural: new ..	151, 156
\@Glsplural@: added accessibility support	66	\GLSaccessstext: new	148, 155
\@Glsymbol@: added accessibility support	69	\glsentryfmt: moved	
\@Glsymbolplural@: added accessibility support	69	\glssetabbrvfmt from	
\@Glstext@: added accessibility support	64	\glsxtrabbrvfmt to here	56
\@glsdesc@: added accessibility support	67	\GlsXtrEnableInitialTagging: new	176
\@glsdescplural@: added accessibility support	68	\glsxtrfieldtitlecase: new	163
\@glsfirst@: added accessibility support	65	\GlsXtrFormatLocationList: new ...	54
\@glsfirstplural@: added accessibility support	66	\glsxtrnewabbrevpresetkeyhook:	
\@Glsname@: added accessibility support	67	new	187
\@Glsplural@: added accessibility support	66	\glsxtrtagfont: new	178
\@Glsymbol@: added accessibility support	68	\KV@printgloss@nonumberlist: added	56
\@Glsymbolplural@: added accessibility support	69	\mfu@checkword@do: added	177
\@Glstext@: added accessibility support	64	\setabbreviationstyle: added check	
\@glsxtr@activate@initialtagging:		for post-definition style switch	203
new	178	0.5.3 (2015-12-09)	
\@glsxtr@do@titlecaps@warn: new ..	178	\@glsxtr@autoindex@at: new	173
\@glsxtr@tag: new	178	\@glsxtr@autoindex@encap: new ...	174
General: fixed typo in glossaries-accsupp		\@glsxtr@autoindex@esc: new	174
and tidied up code to use just one		\@glsxtr@autoindex@level: new ...	174
\@ifpackageloaded	147	\@glsxtr@autoindex@setname: new ..	172
removed \glsxtrabbrvfmt	200	\@glsxtr@doabbreviationsdef: new ..	17
\glossaryentrynumbers: added	54	General: removed	
\Glossentrydesc: added	176	\GlsXtrNoGlsWarningNoAutoMakeMain	
\Glossentryname: added	167	125
\Glossentrysymbol: added	176	\glsdescwidth: added	53
\glossentrysymbol: added	176	\glspagelistwidth: added	53
\GLSaccessdesc: new	151, 156	\glsxtrdoautoindexname: new	171
\GLSaccessdescplural: new ...	152, 156	\glsxtrpostnamehook: new	169
\GLSaccessfirst: new	149, 155	\if@glsxtr@format@override: new ..	171
\GLSaccessfirstplural: new ..	149, 155	\ProvidesGlossariesExtraLang: new	318
\GLSaccesslong: new	153, 157	\RequireGlossariesExtraLang: new	318
\GLSaccesslongpl: new	154, 157	0.5.4 (2015-12-15)	
\Glsaccesslongpl: new	154	\@newglossaryentry@defunitcounters:	
\glsaccesslongpl: new	154	new	101
\GLSaccessname: new	147, 154	\@GLSxtr@p@acrlong@: new	87
\GLSaccessplural: new	148, 155	\@GLSxtr@p@acrlongpl@: new	87
\GLSaccessshort: new	152, 156	\@GLSxtr@p@acrshort@: new	87
\GLSaccessshortpl: new	153, 157	\@GLSxtr@p@acrshortpl@: new	87
\GLSaccesssymbol: new	150, 155	\@GLSxtr@p@long@: new	86
		\@GLSxtr@p@longpl@: new	87
		\@GLSxtr@p@plural@: new	85
		\@GLSxtr@p@short@: new	86
		\@GLSxtr@p@shortpl@: new	86
		\@GLSxtr@p@text@: new	85
		\@GlsXtrEnableOnTheFly: new	50
		\@Glsxtr: new	50
		\@Glsxtr@p@acrlong@: new	87
		\@Glsxtr@p@acrlongpl@: new	87

\@Glsxtr@p@acrshort@: new	87	\glsenableentryunitcount: new ...	103
\@Glsxtr@p@acrshortpl@: new	87	\glshasattribute: added check for	
\@Glsxtr@p@long@: new	86	entry's existence	159
\@Glsxtr@p@longpl@: new	86	\glusifattribute: added check for	
\@Glsxtr@p@plural@: new	85	entry's existence	159
\@Glsxtr@p@short@: new	85	\glspostlinkhook: added existence	
\@Glsxtr@p@shortpl@: new	86	check	179
\@Glsxtr@p@text@: new	85	\Glsxtr: new	50
\@Glsxtrpl: new	51	\glxtr: new	50
\@alt@glshyp@opt: new	81	\glxtrcat: new	50
\@glscalt@hyp@opt: new	81	\glxtrdowrglossaryhook: new	81
\@glscalt@hyp@opt@char: new	81	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	81	new	106
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new	49
new	102	\Glsxtrpl: new	51
\@glsc@local@increment@currunitcount:		\glxtrpl: new	51
new	102	\glxtrpostlocalreset: new	94
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new	94
new	78	\glxtrpostreset: new	94
\@glsxtr: new	50	\glxtrpostunset: new	93
\@glsxtr@addunitcounter: new	101	\glxtrprotectlinks: new	84
\@glsxtr@currunitcount: new	103	\GlsXtrSetAltModifier: new	81
\@glsxtr@ifunitcounter: new	101	\GlsXtrSetDefaultGlsOpts: new	80
\@glsxtr@p@acrlong@: new	87	\glxtrstarflywarn: new	49
\@glsxtr@p@acrlongpl@: new	87	\GlsXtrWarning: new	51
\@glsxtr@p@acrshort@: new	87	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new	87	disables \setacronymstyle	108
\@glsxtr@p@long@: new	86	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new	86	\@glsxtr@autoindexcrossrefs: new .	15
\@glsxtr@p@plural@: new	85	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new	85	new	116
\@glsxtr@p@shortpl@: new	86	\@glsxtr@idx@entrynumberlist: new	118
\@glsxtr@p@text@: new	85	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new	103	new	117
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new	106	new	118
\@glsxtr@unitcountlist: new	101	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new	51	new	117
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new	109
empty see value if not set and added		\makeglossaries: new	110
'see' to field key map	41	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	49	\glxtrdiscardperiod: added check	
\cGlsformat: added	100	for first use	180
\cGlsformat: added	100	short-desc: fixed typo in	
\cGlsplformat: added	101	\glxtrinlinfullformat and	
\cGlsplformat: added	100	added missing second argument ...	217
\glstdisablehyper: added	83	1.02 (2016-04-25)	
\glstdohyperlink: added	82	\@glsxtr@current@style: new	52
\glstdonohyperlink: added	84	\Glsfmtfull: new	317

\glsfmtfull: new	317	\@GLSdescplural@: set abbreviation and regular format	68
\Glsfmtfullpl: new	318	\@GLSfirst@: set abbreviation format ..	65
\glsfmtfullpl: new	318	\@GLSfirstplural@: set abbreviation and regular format	67
\Glsfmtlong: new	316	\@GLSname@: set abbreviation and regular format	67
\glsfmtlong: new	316	\@GLSplural@: set abbreviation and regular format	66
\Glsfmtlongpl: new	317	\@GLSsymbol@: set regular format	69
\glsfmtlongpl: new	316	\@GLSsymbolplural@: set regular format	69
\Glsxtrheadfull: new	311	\@GLStext@: set abbreviation and regular format	64
\glsxtrheadfull: new	310	\@GLSuseri@: set regular format	70
\Glsxtrheadfullpl: new	311	\@GLSuserii@: set regular format	70
\glsxtrheadfullpl: new	310	\@GLSuseriii@: set regular format	70
\Glsxtrheadlong: new	309	\@GLSuseriv@: set regular format	71
\glsxtrheadlong: new	308	\@GLSuservi@: set regular format	71
\Glsxtrheadlongpl: new	310	\@GLSdesc@: set abbreviation and regular format	68
\glsxtrheadlongpl: new	309	\@GLSdescplural@: set abbreviation and regular format	68
\Glsxtrtitlefull: new	311	\@Glsfirst@: set abbreviation and regular format	65
\glsxtrtitlefull: new	310	\@Glsfirstplural@: set abbreviation and regular format	66
\Glsxtrtitlefullpl: new	312	\@Glsname@: set abbreviation and regular format	67
\glsxtrtitlefullpl: new	311	\@Glsplural@: set abbreviation and regular format	66
\Glsxtrtitlelong: new	309	\@Glsymbol@: set regular format	69
\glsxtrtitlelong: new	309	\@Glsymbolplural@: set regular format	69
\Glsxtrtitlelongpl: new	310	\@Glstext@: set abbreviation and regular format	64
\glsxtrtitlelongpl: new	309	\@Glsuseri@: set regular format	69
\ifglsxtrinsertinside: new	206	\@Glsuserii@: set regular format	70
postfootnote: added redef of		\@Glsuseriii@: set regular format	70
\glsxtrsetupfulldefs	213	\@Glsuseriv@: set regular format	70
stylemods: new	21	\@Glsuservi@: set regular format	71
1.03 (2016-04-27)		\@GLSdesc@: set abbreviation and regular format	68
\@GLSfirstplural@: bug fix: misspelt cs name	67	\@GLSdescplural@: set abbreviation and regular format	66
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	66	\@Glsfirst@: set abbreviation and regular format	65
\@Glsfirstplural@: bug fix: misspelt cs name	66	\@Glsfirstplural@: set abbreviation and regular format	66
\@Glsplural@: fixed bug \@Glsplural@ should be redefined not \@Glsplural	66	\@Glsname@: set abbreviation and regular format	67
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	66	\@Glsplural@: set abbreviation and regular format	66
\glsxtrtitlelongpl: bug fix: changed \glsxtrlong to \glsxtrlongpl ..	309	\@Glsymbol@: set regular format	69
\glsxtrtitleshortpl: bug fix: changed \glsxtrshort to \glsxtrshortpl	303	\@Glsymbolplural@: set regular format	69
1.04 (2015-04-30)		\@Glstext@: set abbreviation and regular format	64
short-em-footnote: renamed from “footnote-em”	269	\@Glsuseri@: set regular format	69
1.04 (2016-05-02)		\@Glsuserii@: set regular format	70
\@glsxtrpostloctag: new	55	\@Glsuseriii@: set regular format	70
\@GLSdesc@: set abbreviation and regular format	68	\@Glsuseriv@: set regular format	70
		\@Glsuservi@: set regular format	71
		\@Glsdesc@: set abbreviation and regular format	67
		\@Glsdescplural@: set abbreviation and regular format	68
		\@Glsfirst@: set abbreviation and regular format	65

\@glsfirstplural@: set abbreviation and regular format	66	\glsxtruserparen: new	273
\@glsname@: set abbreviation and regular format	67	\glsxtrusersuffix: new	274
\@glsplural@: set abbreviation and regular format	66	\GlsXtrWarnDeprecatedAbbrStyle: new	205
\@glsymbol@: set regular format	68	short-em-long-em: new	257
\@glsymbolplural@: set regular format	69	short-em-long-em-desc: new	258
\@glstext@: set abbreviation and regular format	64	short-em-nolong: new	260
\@glsxtr@deprecated@abbrstyle: new	205	short-em-nolong-desc: new	261
\@glsxtr@do@style: new	22	short-em-postfootnote: renamed from “postfootnote-em”	271
\@glsxtr@doloctag: new	56	short-footnote: new	213
\@glsxtr@idx@entrynumberlist: switched from \let to \newcommand	118	short-long-user: new	280
\@glsxtr@pagetag: new	55	short-long-user-desc: new	281
\@glsxtr@pagetag: new	55	short-nolong: new	216
\@glsxtr@preloctag: new	56	short-nolong-desc: new	218
\@glsxtr@postloctag: new	56	short-postfootnote: new	215
\@glsxtr@preloctag: new	55	short-sc-footnote: renamed from “footnote-sc”	233
\glossentrydesc: added glossdescfont attribute check	163	short-sc-nolong: new	228
\Glossentryname: added glossnamefont attribute check	167	short-sc-nolong-desc: new	229
\glossentryname: added glossnamefont attribute check	165	short-sc-postfootnote: renamed from “postfootnote-sc”	235
moved post name hook inside condition	167	short-sm-footnote: renamed from “footnote-sm”	248
\glsabbrvmfont: new	251	short-sm-nolong: new	242
\glsabbrvuserfont: new	273	short-sm-nolong-desc: new	244
\glsfirstabbrvmfont: new	251	short-sm-postfootnote: renamed from “postfootnote-sm”	249
\glsfirstabbrvuserfont: new	273	\letabbreviationstyle: new	205
\glsfirstlongemfont: new	251	\newabbreviationstyle: bug fix: corrected test for existence	204
\glsfirstlonguserfont: new	274	long-em-noshort-em: new	264
\glsifnotregularcategory: new ...	160	long-em-noshort-em-desc: new ...	267
\glslongdefaultfont: new	189	long-em-short-em: new	253
\glslongemfont: new	252	long-em-short-em-desc: new	254
\glslongfont: new	189	long-noshort: new	222
\glslonguserfont: new	273	long-noshort-desc: new	221
\glsxtrassignfieldfont: new	64	long-noshort-em: renamed from “long-em”	262
\GlsXtrEnablePreLocationTag: new .	54	long-noshort-em-desc: renamed from “long-desc-em”	266
\glsxtrfirstscfont: new	223	long-noshort-sc: renamed from “long-sc”	230
\glsxtrfirstsmfont: new	237	long-noshort-sc-desc: renamed from “long-desc-sc”	232
\glsxtrlongshortdescsort: new ...	208	long-noshort-sm: renamed from “long-sm”	244
\glsxtrpostnamehook: added category check	169	long-noshort-sm-desc: renamed from \long-desc-sm	246
\glsxtrregularfont: new	57		
\glsxtruserfield: new	273		

long-short-user: new	274	docdef option changed to choice	14
long-short-user-desc: new	280	\glxtr@usesee: new	41
\renewabbreviationstyle: new	204	\glxtrusesee: new	41
style: new	22	\glxtruseseeformat: new	41
1.05 (2016-06-10)		\if@glxtrdocdefrestricted: new ..	15
\eglssetwidest: new	372	1.07 (2016-08-15)	
\glFindWidestAnyName: new	375	\@@glxtrp: new	88
\glFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new	380	nohyperfirst attribute	65
\glFindWidestAnyNameSymbol: new	378	\@GLSfirstplural@: added check for	
\glFindWidestAnyNameSymbolLocation:		nohyperfirst attribute	67
new	379	\@GLSxtrp: new	89
\glFindWidestLevelTwo: new	376	\@Glsfirst@: added check for	
\glFindWidestUsedAnyName: new ..	375	nohyperfirst attribute	65
\glFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new	380	nohyperfirst attribute	66
\glFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new	88
new	377	\@gls@preglossaryhook: added	
\glFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts	178
new	378	\@glfirst@: added check for	
\glFindWidestUsedLevelTwo: new .	375	nohyperfirst attribute	65
\glFindWidestUsedTopLevelName:		\@glfirstplural@: added check for	
new	374	nohyperfirst attribute	66
\glfirstlongfootnotefont: new ..	211	\@glxtrinmark: new	300
\glsgetwidestname: new	374	\@glxtrnotinmark: new	300
\glsgetwidestsubname: new	374	\@glxtrp: new	88
\glslongfootnotefont: new	211	\@glxtrp@opt: new	87
\glxtrAltTreeIndent: new	372	\glossxtrsetpopts: new	88
\glxtralttreeInit: new	372	\glsp: new	90
\glxtrAltTreePar: new	372	\glsp: new	90
\glxtrAltTreeSetHangIndent: new	381	\glxtr@entry@p: new	89
\glxtrAltTreeSetSubHangIndent:		\glxtrabbrvfootnote: new	211
new	382	\glxtrchecknohyperfirst: new	65
\glxtralttreeSubSymbolDescLocation:		\glxtrfieldtitlecasecs: new	163
new	372	\glxtrifinmark: new	300
\glxtralttreeSymbolDescLocation:		\GLSxtrp: new	91
new	371	\Glsxtrp: new	90
\glxtrComputeTreeIndent: new ...	381	\glxtrp: new	89
\glxtrComputeTreeSubIndent: new	381	\glxtrsetpopts: new	88
\glxtrtreetopindent: new	372	short-long-desc: added text key	210
short-em-long: fixed incorrect font used		fixed misspelling of \glabbrvfont in	
by long form	256	plural key	210
\xglsetwidest: new	373	long-short-desc: added missing text	
1.06 (2016-06-18)		key	208
\@glsoifexistsorwarn: new	15	fixed misspelling of \glabbrvfont .	208
\@glxtr@docdefval: new	14	footnote: changed first forms to use	
\@glxtr@usesee: new	41	\glfirstlongfootnotefont ...	211
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document	27	from first keys	213

switched from \glsfirstlongfont to \glsfirstlongfootnotefont ...	214		
\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse	109		
1.08 (2016-12-13)			
\@glsxtr@record: new	8		
\@GLS@: added \glsxtr@record	58		
\@GLSpl@: added \glsxtr@record ...	58		
\@Gls@: added \glsxtr@record	58		
\@GLspl@: added \glsxtr@record ...	58		
\@gls@: added \glsxtr@record	57		
\@gls@@link@: added \glsxtr@record	59		
\@gls@field@link: added \glsxtr@record	57		
\@gls@saveentrycounter: new	27		
\@glsdisp: added \glsxtr@record ..	58		
\@glspl@: added \glsxtr@record ...	58		
\@glsxtr@dorecord: new	10		
\@glsxtr@err@undefaction: new	6		
\@glsxtr@record: new	7		
\@glsxtr@warn@onexistsordo: new ...	6		
\@glsxtr@warn@undefaction: new	6		
\@print@unsrt@glossary: new	133		
General: added record package option ...	13		
\glsadd: added \glsxtr@record	63		
\glsdoifexists: now defines \glslabel	39		
\glsxtr@do@wrglossary: new	27		
\glsxtr@addloclistfield: new	12		
\glsxtr@indexonly@saveentrycounter: new	12		
\glsxtr@record: new	130		
\glsxtr@resource: new	128		
\glsxtr@saveentrycounter: new	27		
\glsxtr@setup@record: new	12		
\glsxtr@assignfieldfont: added check for existence	64		
\glsxtr@resourcefile: new	127		
\printunsrtglossaries: new	133		
\printunsrtglossary: new	132		
1.09 (2016-12-16)			
\@glsxtr@gettype: new	116		
\@glsxtr@mixed@assign@sortkey: new	116		
\@printglossary: redefined to save options	115		
\glsxtr@makeglossaries: new	116		
		1.10 (2016-12-17)	
		\@GLSpl@: fixed bug caused by typo in command name	58
		1.11 (2017-01-19)	
		\@glsxtr@do@redef@forglsentries: new	6
		\@glsxtr@noidx@do: new	136
		\@glsxtr@redef@forglsentries: new .	6
		\@glsxtr@shortcutsval: new	20
		\@glsxtr@unsrt@getgrouptitle: new	135
		\@print@noidx@glossary: added redefinition	120
		\glsxtr@addloclistfield: added group key	13
		added location key	12
		\glsxtr@fields: new	128
		\glsxtr@linkprefix: new	128
		\glsxtr@org@newignoredglossary: new	36
		\glsxtr@s@newignoredglossary: new	37
		\glsxtr@shortcutsval: new	128
		\glsxtr@texencoding: new	128
		\glsxtr@writefields: new	129
		\GlsXtrLoadResources: new	128
		\glsxtrpageref: new	34
		\glsxtr@resourcefile: changed extension to .glstex	127
		\newignoredglossary: added starred version	36
		1.12 (2017-02-03)	
		\@glsxtr@recordcounter: new	11
		\@gls@preglossaryhook: check for definition	178
		\@glsxtr@counterrecordhook: new .	130
		\@glsxtr@display@loc: new	120
		\@glsxtr@docounterrecord: new ...	130
		\@glsxtr@longnewglossaryentry: new	35
		\@glsxtr@noop@recordcounter: new .	11
		\@glsxtr@op@recordcounter: new ...	11
		\@glsxtr@provide@storagekey: new .	28
		\@glsxtr@s@longnewglossaryentry: new	35
		\@glsxtrentryfmt: new	30
		\@glsxtrindexaliased: new	79
		\@glsxtrsetaliasnoindex: new	79
		\@newglossaryentryposthook: added check for alias key	45
		\@no@glsxtrindexaliased: new	79

\@printunsrtglossary: new	132	\GlsXtrLoadResources: removed	
General: added target key to printgloss		restriction on only one per document	128
family	115	\glxtrlocrangefmt: new	122
\apptoglossarypreamble: new	34	\glxtrpostlongdescription: new	36
\csGlsXtrLetField: new	33	\glxtrprovidestoragekey: new	28
\eGlsXtrSetField: new	34	\GlsXtrRecordCounter: new	130
\gGlsXtrSetField: new	33	\glxtrresourcecount: new	128
\glsdohyperlink: added check for alias		\glxtrresourcefile: added catcode	
field	82	change for @	127
\glsnoidxdisplayloc: added		\glxtrsetaliasnoindex: new	79
redefinition	120	\GlsXtrSetField: new	33
\glissettoctitle: added patch	37	\glxtrsetfieldifexists: new	33
\glxtr@counterrecord: new	130	\glxtrunsrtdo: new	135
\glxtr@langtag: new	128	\GlsXtrusefield: new	33
\glxtr@newabbreviation: new	185	\glxtrusefield: new	32
\glxtr@org@newignoredglossary:		short-postlong-user: new	277
Added check for existence	36	short-postlong-user-desc: new	279
\glxtr@pluralsuffixes: new	128	\longnewglossaryentry: added starred	
\glxtr@provideignoredglossary:		version	35
new	38	long-postshort-user: new	275
\glxtr@s@newignoredglossary:		long-postshort-user-desc: new	277
Added check for existence	37	postdot: new	16
\glxtr@s@provideignoredglossary:		\pretoglossarypreamble: new	34
new	38	\print@noop@unsrtglossaryunit:	
\glxtrabbrvpluralsuffix: new	189	new	135
\glxtralias: new	45	\print@op@unsrtglossaryunit: new	135
\glxtrcopytoglossary: new	39	\printunsrtglossary: added starred	
\glxtrdeffield: new	33	form	132
\glxtrdisplayendloc: new	121	\printunsrtglossaryhandler: new	134
\glxtrdisplayendloohook: new	121	\printunsrtglossaryunit: new	12
\glxtrdisplayingleloc: new	121	\printunsrtglossaryunitsetup: new	135
\glxtrdisplaystartloc: new	121	\provideignoredglossary: new	38
\glxtrdeffield: new	33	\s@glxtr@provide@storagekey: new	29
\glxtrentryfmt: new	30	\s@printunsrtglossary: new	132
\glxtrfieldddolistloop: new	31	\xGlsXtrSetField: new	34
\glxtrfieldforlistloop: new	31		
\glxtrfielddifylist: new	31	1.13 (2017-02-07)	
\glxtrfieldlistadd: new	31	\@glldisp: removed	
\glxtrfieldlistadd: new	31	\@glxtr@org@glldisp	58
\glxtrfieldlistgadd: new	31	\glxtrsetaliasnoindex: switched to	
\glxtrfieldlistxadd: new	31	\providecommand	79
\glxtrfieldxifylist: new	31	1.14 (2017-04-18)	
\glxtrfmt: new	29	\@gls@link: added redefinition	61
\GlsXtrFmtDefaultOptions: new	29	\@gls@noidx@getgrouptitle: new	118
\GlsXtrFmtField: new	29	\@gls@removespaces: new	122
\glxtrifkeydefined: new	28	\@glxtr@do@automake@err: new	130
\glxtrindexaliased: new	79	\@glxtr@org@gloautosee: new	25
\GlsXtrLetField: new	33	\@glxtr@record: added third arg	7
\GlsXtrLetFieldToField: new	33	\@glxtr@recordsee: new	11

General: added \glsadd option		1.16 (2017-06-15)	
theHvalue	63	\@glo@autosee: added redefinition	26
added \glsadd option thevalue	63	\@gls@noidx@getgrouptitle: fixed	
\glsdisablehyper: added redefinition .	83	bug	118
\glsenableentrycount: fixed		\@glsxtr@addunusedxrefs: added	
assignment of \@cGls@	96	check for seealso field	46
\glsenableentryunitcount: fixed		\@glsxtr@checkgroup: use \csuse	
assignment of \@cGls@	104	instead of \csname	136
\glsnavigation: new	119	\@glsxtr@dorecordnodefer: new	11
\glsxtr@org@getgrouptitle: new ..	119	\@print@unsrt@glossary: corrected	
\glsxtr@recordsee: new	7	misspelt command	133
\glsxtr@writefields: added check for		\@printunsrt@glossary@handler:	
automake	130	new	134
\glsxtrdisplayendloc: added check		General: added check for	
for empty format	121	\@gls@setupsort@none	14
\glsxtrgetgrouptitle: new	119	\@gls@checkseeallowed: added	
\glsxtrinitwrgloss: new	59	redefinition	26
\glsxtrlocationhyperlink: new ...	122	\@glsxtr@writefields: added	
\glsxtrsetgrouptitle: new	119	\providecommand lines	129
\glsxtrsupphypernumber: new	122	\@glsxtrautoindex: new	172
\ifglsxtrwrglossbefore: new	59	\@glsxtrautoindexassignsort: new .	172
1.15 (2017-05-10)		\@glsxtrautoindexentry: new	172
\@glsxtr@dorecord: corrected		\@glsxtrindexseealso: new	42
premature expansion of \@glslocref	10	\@glsxtrseealsolabels: new	45
short-em-long-em: fixed spelling of		\@glsxtrseelist: new	42
\glsabbrvfont	257	\@glsxtruseseealso: new	42
short-long: fixed spelling of		\@glsxtruseseealsoformat: new	42
\glsabbrvfont	209	\@seealso: new	42
short-long-user: fixed spelling of		autoseeindex: new	16
\glsabbrvfont	281	1.17 (2017-08-09)	
short-postlong-user: fixed spelling of		\@glsxtr@mark@wordseps: new	184
\glsabbrvfont	277	\@glsxtr@markwordseps: new	184
short-postlong-user-desc: fixed		\@glsxtr@noidx@displaynumberlist:	
spelling of \glsabbrvfont	279	replace hard-coded ?? with	
long-em-short-em: fixed spelling of		\glsxtrundeftag	117
\glsabbrvfont	254	\@glsxtr@noidx@entrynumberlist:	
long-postshort-user: fixed spelling of		replace hard-coded ?? with	
\glsabbrvfont	275	\glsxtrundeftag	118
long-postshort-user-desc: fixed		\@glsxtr@noidx@numberlistloop:	
spelling of \glsabbrvfont	277	replace hard-coded ?? with	
long-short: fixed spelling of		\glsxtrundeftag	117
\glsabbrvfont	207	\@glsxtrifhyphenstart: new	282
long-short-user: fixed spelling of		General: removed some inconsistencies	
\glsabbrvfont	274	in the abbreviation styles	206
footnote: fixed spelling of		\glsabbrvhyphenfont: new	283
\glsabbrvfont	211	\glsabbrvonlyfont: new	296
postfootnote: fixed spelling of		\glsabbrvscfont: new	223
\glsabbrvfont	213	\glsabbrvsmfont: new	237

<code>\glsabbrvuserfont</code> : initialised to default font	273	short-long-user-desc: corrected first forms	282
<code>\glsfirstabbrvhyphenfont</code> : new	283	short-nolong-desc-noreg: new	218
<code>\glsfirstabbrvonlyfont</code> : new	296	short-nolong-noreg: new	216
<code>\glsfirstabbrvscfont</code> : new	223	long-em-noshort-em-desc-noreg: new	269
<code>\glsfirstabbrvsmfont</code> : new	237	long-em-noshort-em-noreg: new	265
<code>\glsfirstlonghyphenfont</code> : new	283	long-hyphen-noshort-desc-noreg: new	285
<code>\glsfirstlongonlyfont</code> : new	296	long-hyphen-postshort-hyphen: new	289
<code>\glslonghyphenfont</code> : new	283	long-hyphen-postshort-hyphen-desc: new	290
<code>\glslongonlyfont</code> : new	296	long-hyphen-short-hyphen: new	283
<code>\glslonguserfont</code> : initialised to default font	273	long-hyphen-short-hyphen-desc: new	284
<code>\glsxtr@newabbreviation</code> : added <code>\glsxtrorgshort</code> and <code>\glsxtrorglong</code>	185	long-noshort-desc-noreg: new	221
<code>\GlsXtrDefineAcShortcuts</code> : new	18	long-noshort-noreg: new	222
<code>\glsxtrgenabbrvfmt</code> : added check for <code>\ifglsxtrininsertinside</code>	200	long-only-short-only: new	297
<code>\glsxtrhyphensuffix</code> : new	283	long-only-short-only-desc: new	298
<code>\glsxtrifhyphenstart</code> : new	282	long-short-user-desc: corrected first forms	280
<code>\glsxtrlonghyphen</code> : new	288	1.18 (2017-08-10)	
<code>\glsxtrlonghyphennoshort</code> : new	285	stylemods: changed default value to "default"	21
<code>\glsxtrlonghyphenshort</code> : new	283	1.19 (2017-09-09)	
<code>\glsxtrlongshortdescname</code> : new	208	<code>\@glsxtr@defaultnumberformat</code> : new	7
<code>\glsxtronlydescname</code> : new	298	<code>\@glsxtr@dorecord</code> : Use <code>\@glsrecordloc</code> instead of <code>\@glsloc</code>	10
<code>\glsxtronlydescsort</code> : new	298	<code>\@glsxtr@dorecordnodefer</code> : Use <code>\theglentrycounter</code> for the location rather than <code>\@glsloc</code>	11
<code>\glsxtronlysuffix</code> : new	297	<code>\@glsxtr@record@setting</code> : new	13
<code>\glsxtrparen</code> : new	187	<code>\@glsxtr@record@setting@alsoindex</code> : new	13
<code>\glsxtrposthyphenlong</code> : new	293	<code>\@glsxtrifhasfield</code> : new	32
<code>\glsxtrposthyphenshort</code> : new	288	General: added <code>\glslink</code> option <code>theHvalue</code>	60
<code>\glsxtrposthyphensubsequent</code> : new	288	added <code>\glslink</code> option <code>thevalue</code>	60
<code>\glsxtrshortdescname</code> : new	217	<code>\glsxtr@writefields</code> : removed double-quotes around <code>\jobname</code>	130
<code>\glsxtrshorthyphen</code> : new	293	<code>\glsxtrdoautoindexname</code> : changed format test	172
<code>\glsxtrshorthyphenlong</code> : new	291	<code>\glsxtrhyperlink</code> : new	83
<code>\glsxtrshortlongdescname</code> : new	210	<code>\glsxtrifhasfield</code> : new	32
<code>\glsxtrshortlongdescsort</code> : new	210	<code>\GlsXtrSetDefaultNumberFormat</code> : new	7
<code>\Glsxtrsubsequentfmt</code> : new	203	<code>\s@glsxtrifhasfield</code> : new	32
<code>\glsxtrsubsequentfmt</code> : new	202		
<code>\Glsxtrsubsequentplfmt</code> : new	203		
<code>\glsxtrsubsequentplfmt</code> : new	202		
<code>\glsxtrword</code> : new	184		
<code>\glsxtrwordsep</code> : new	184		
short-hyphen-long-hyphen: new	291		
short-hyphen-long-hyphen-desc: new	292		
short-hyphen-postlong-hyphen: new	294		
short-hyphen-postlong-hyphen-desc: new	296		

1.20 (2017-09-11)		
\@glstrhypernameprefix: new	116	
\glsdohypertarget: added redefinition	116	
\printunsrtglossaryunitsetup:		
switched from redefining		
\glolinkprefix to		
\@glstrhypernameprefix	135	
1.21 (2017-11-03)		
\@glstr@record: added check for		
default options	9	
\@glstrwrglossmark: new	24	
\glslink: changed \let to \def	84	
\@glstr@checkgroup: new	136	
\@glstr@defpostpunc: new	16	
\@glstr@do@record@wrglossary:		
new	8	
\@glstr@dosee@alsoindex@glossary:		
new	25	
\@glstr@doseeglossary: new	25	
\@glstr@noidx@do: removed code		
dealing with the group	137	
\@glstr@record@setting@off: new	13	
\@glstr@record@setting@only: new	13	
\@glstr@rglstrigger@record: new	141	
\@glstrglossentry: new	131	
\@glstrnewglsl: new	138	
\@glstrsetaliasnoindex: changed to		
use \glstrifhasfield instead of		
\ifglshasfield	79	
\@glstrwrglossmark: new	24	
\@rGLS: new	143	
\@rGLS@: new	143	
\@rGLSpl: new	144	
\@rGLSpl@: new	144	
\@rGls: new	143	
\@rGls@: new	143	
\@rGlspl: new	143	
\@rGlspl@: new	143	
\@rgls: new	142	
\@rgls@: new	142	
\@rglspl: new	142	
\@rglspl@: new	142	
General: adjusted mcolalttree	387	
ac	21	
modified index to remove hard coded		
\space	367	
modified list to remove hard coded		
\space	357	
		moved conditional outside of
		\glsgroupskip
		new
		redefined altlistgroup to discourage
		breaks after group headings
		redefined altlisthypergroup to
		discourage breaks after group
		headings
		redefined alttreegroup to discourage
		breaks after group headings
		redefined alttreehypergroup to
		discourage breaks after group
		headings
		redefined indexgroup to discourage
		breaks after group headings
		redefined indexhypergroup to
		discourage breaks after group
		headings
		redefined listgroup to discourage
		breaks after group headings
		redefined listhypergroup to
		discourage breaks after group
		headings
		redefined mcolalttreegroup to
		discourage breaks after group
		headings
		redefined mcolalttreehypergroup to
		discourage breaks after group
		headings
		redefined mcolalttreespannav to
		discourage breaks after group
		headings
		redefined mcolindexgroup to
		discourage breaks after group
		headings
		redefined mcolindexhypergroup to
		discourage breaks after group
		headings
		redefined mcolindexspannav to
		discourage breaks after group
		headings
		redefined mcoltreegroup to
		discourage breaks after group
		headings
		redefined mcoltreehypergroup to
		discourage breaks after group
		headings
		redefined mcoltreenamegroup to
		discourage breaks after group

headings	386	\glxtrbookindexbetween: new	391
redefined		\glxtrbookindexbookmark: new ...	391
mcoltreenonamehypergroup to		\glxtrbookindexcols: new	390
discourage breaks after group		\glxtrbookindexcolspread: new ..	391
headings	386	\glxtrbookindexfirstmark: new ..	395
redefined mcoltreenonamespannav to		\glxtrbookindexfirstmarkfmt: new	395
discourage breaks after group		\glxtrbookindexformatheader: new	391
headings	387	\glxtrbookindexgroupskip: new ..	391
redefined mcoltreepspannav to		\glxtrbookindexlastmark: new ...	395
discourage breaks after group		\glxtrbookindexlastmarkfmt: new	395
headings	385	\glxtrbookindexmarkentry: new ..	395
redefined treegroup to discourage		\glxtrbookindexname: new	390
breaks after group headings	369	\glxtrbookindexparentchildsep:	
redefined treehypergroup to		new	390
discourage breaks after group		\glxtrbookindexparentschildsep:	
headings	370	new	390
redefined treenonamegroup to		\glxtrbookindexprelocation: new	390
discourage breaks after group		\glxtrbookindexsubatendgroup:	
headings	371	new	391
redefined treenonamehypergroup to		\glxtrbookindexsubbetween: new ..	391
discourage breaks after group		\glxtrbookindexsubname: new	390
headings	371	\glxtrbookindexsubprelocation:	
debug: new	24	new	390
\gglssetwidest: new	372	\glxtrbookindexsubsubatendgroup:	
\glsdisablehyper: added check for		new	391
existence	83	\glxtrbookindexsubsubbetween:	
changed to use \def rather than \let .	83	new	391
\glsenablehyper: changed to use \def		\glxtrbookindexthepage: new	394
rather than \let	83	\glxtrdetoklocation: new	140
\Glsfmtname: new	313	\glxtrenablerecordcount: new ...	140
\glsfmtname: new	313	\glxtrglossentry: new	131
\glshex: new	319	\glxtrgroupfield: new	136
\glslistchildpostlocation: new ..	357	\Glsxtrheadname: new	305
\glslistchildprelocation: new ...	357	\glxtrheadname: new	304
\glslistprelocation: new	357	\GlsXtrIfFieldEqStr: new	34
\glsnahyperlink: patched	81	\glxtriflabelinlist: new	134
\glssseeitemformat: new	42	\glxtrifrecordtrigger: new	141
\glsshowsorttarget: new	25	\glxtrindexseealso: added check	
\glstreechildprelocation: new ...	367	that the entry exists	43
\glstreeprelocation: new	367	\glxtrtrinityhyperoutside: new	60
\glstriggerrecordformat: new	142	\GlsXtrLocationRecordCount: new ..	140
\glseuseabbrvfont: new	200	\glxtrnewgls: new	137, 138
\glseuselongsfont: new	200	\glxtrnewGLSlike: new	139
\glxtr@do@alsoindex@wrglossary:		\glxtrnewglslike: new	139
new	8	\glxtrnewrgls: new	139
\glxtr@org@@do@wrglossary: new ..	27	\glxtrnewrGLSlike: new	139
\glxtr@org@dohyperlink: new	81	\glxtrnewrglslike: new	139
\glxtr@setbookindexmark: new ...	395	\glxtrprelocation: new	356, 390
\glxtrbookindexatendgroup: new ..	391	\GlsXtrRecordCount: new	139

\glxtrrecordtriggervalue: new ..	140	\glseeitemformat: switched check	
\glxtrresourcefile: now disables		from regular to short	42
record key	127	\glxtr@setaccessdisplay: new ...	169
\glxtrresourceinit: new	128	\glxtr@writefields: provide	
\GlsXtrSetRecordCountAttribute:		\glxtr@record in aux file	129
new	140	\glxtractivatenopost: new	114
\glxtrtitlename: new	305	\glxtrbookindexprelocation:	
\glxtrtitleorpdforheading: new .	301	removed check for no post dot	390
\GlsXtrTotalRecordCount: new	139	\glxtrglossentryother: new	131
\glxtrwrglossmark: new	24	\glxtrnopostpunc: new	115
short-em: new	259	1.23 (2017-11-12)	
short-sc: corrected first letter		\@@glxtrfmt: added check for indexing	30
uppercasing	227	added grouping	29
short-sm: corrected first letter		new	29
uppercasing	242	\@glxtr@nopostpunc@postdesc: new	115
\ifglxtr@hyperoutside: new	60	\@glxtr@restore@postpunc: new ..	115
all: new	355	\@glxtrentryfmt: fixed missing label	
nolong-short: new	218	argument	30
nolong-short-em: new	261	\@glxtrfmt: new	29
nolong-short-noreg: new	219	\eglsupdatewidest: new	373
nolong-short-sc: new	229	\gglupdatewidest: new	373
nolong-short-sm: new	244	\glsupdatewidest: new	373
nopostdot: new	16	\GlsXtrDefineAbbreviationShortcuts:	
postpunc: new	16	changed \newabbr definition to use	
\printunsrtglossaryentryprocesshook:		\providecommand	18
new	134	\GlsXtrDefineAcShortcuts: changed	
\printunsrtglossarypredoglossary:		\newabbr definition to use	
new	134	\providecommand	19
\rGLS: new	143	\glxtrfmtdisplay: new	30
\rGLs: new	142	\glxtrifcustomdiscardperiod: new	179
\rgls: new	142	\GlsXtrIfFieldUndef: new	32
\rGLSformat: new	145	\glxtrrestorepostpunc: new	115
\rGLsformat: new	144	\s@glxtrfmt: new	29
\rglsformat: new	144	\s@glxtrfmt: new	29
\rGLSpl: new	144	\xglupdatewidest: new	373
\rGLspl: new	143	1.24 (2017-11-14)	
\rglspl: new	142	\glsadd: added \@gls@setsort	63
\rGLSplformat: new	145	\glxtrforcsvfield: new	32
\rGLsplformat: new	144	\glxtrlocalsetgrouptitle: new ..	119
\rglsplformat: new	144	1.25 (2017-11-14)	
\s@glxtrifhasfield: switched from		\glxtrbookindexmulticolseenv: new	391
\ifdef to \ifundef	32	1.25 (2017-11-24)	
1.22 (2017-11-08)		\glsextrapostnamehook: new	169
\@glxtr@nopostpunc: new	115	\glxtrfootnotename: new	211
\@glxtr@orgprintglossary: changed		\glxtrlongnoshortdescname: new .	220
explicit \let for \nopostdesc to		\glxtrlongnoshortname: new	222
\glxtractivatenopost	114	\glxtrlongshortname: new	206
\@glxtrglossentryother: new	132	\glxtrlongshortuserdescname: new	276
\glossentrynameother: new	170	\glxtronlyname: new	297

\glxtrpostlinkAddDescOnFirstUse: changed to use \glxtrparen	180	\glxtrGeneralLatinVIrules: new ..	333
\glxtrpostlinkAddSymbolOnFirstUse: changed to use \glxtrparen	180	\glxtrGeneralLatinVrules: new ..	332
\glxtrshortlongname: new	208	\glxtrgeneralpuncIIrules: new ..	329
\glxtrshortlonguserdescname: new	279	\glxtrgeneralpuncIrules: new ...	328
\glxtrshortnolongname: new	215	\glxtrgeneralpuncrules: new	328
1.26 (2018-01-05)		\glxtrhyphenrules: new	327
\@glxtr@do@inc@linkcount: new ..	145	\glxtrLatinA: new	335
\glslinkpresetkeys: new	60	\glxtrLatinAA: new	337
\glxtr@inc@linkcount: new	60	\glxtrLatinAELigature: new	337
\GlsXtrEnableLinkCounting: new ..	146	\glxtrLatinE: new	335
\GlsXtrIfLinkCounterDef: new	146	\glxtrLatinEszettSs: new	336
\glxtrinclinkcounter: new	146	\glxtrLatinEszettSz: new	337
\GlsXtrLinkCounterName: new	146	\glxtrLatinEth: new	337
\GlsXtrLinkCounterValue: new	146	\glxtrLatinH: new	335
\GlsXtrTheLinkCounter: new	146	\glxtrLatinI: new	335
1.27 (2018-02-26)		\glxtrLatinInsularG: new	337
\@glxtrdialecthook: new	27	\glxtrLatinK: new	335
General: added		\glxtrLatinL: new	335
glossaries-extra-bib2gls.sty	319	\glxtrLatinLslash: new	338
\Alpha: new	321	\glxtrLatinM: new	336
\Beta: new	321	\glxtrLatinN: new	336
\Chi: new	322	\glxtrLatinO: new	336
\Digamma: new	322	\glxtrLatinOELigature: new	337
\Epsilon: new	322	\glxtrLatinOslash: new	337
\Eta: new	322	\glxtrLatinP: new	336
\glxtr@loaddialect: new	319	\glxtrLatinS: new	336
\glxtrBasicDigitrules: new	352	\glxtrLatinSchwa: new	336
\glxtrcombiningdiacriticIIIrules: new	326	\glxtrLatinT: new	336
\glxtrcombiningdiacriticIIrules: new	326	\glxtrLatinThorn: new	337
\glxtrcombiningdiacriticIrules: new	325	\glxtrLatinWynn: new	337
\glxtrcombiningdiacriticIVrules: new	327	\glxtrLatinX: new	336
\glxtrcombiningdiacriticrules: new	325	\glxtrMathGreekIIrules: new	344
\glxtrcontrolrules: new	324	\glxtrMathGreekIrules: new	343
\glxtrcurrencyrules: new	329	\glxtrMathItalicAlpha: new	348
\glxtrdigitrules: new	351	\glxtrMathItalicBeta: new	348
\glxtrfractionrules: new	352	\glxtrMathItalicChi: new	351
\glxtrGeneralLatinIIIrules: new	331	\glxtrMathItalicDelta: new	348
\glxtrGeneralLatinIIrules: new ..	330	\glxtrMathItalicEpsilon: new ...	348
\glxtrGeneralLatinIrules: new ..	329	\glxtrMathItalicEta: new	349
\glxtrGeneralLatinIVrules: new ..	331	\glxtrMathItalicGamma: new	348
\glxtrGeneralLatinVIIrules: new	334	\glxtrMathItalicGreekIIrules: new	339
\glxtrGeneralLatinVIIrules: new	333	\glxtrMathItalicGreekIrules: new	339
		\glxtrMathItalicIota: new	349
		\glxtrMathItalicKappa: new	349
		\glxtrMathItalicLambda: new	349
		\glxtrMathItalicLowerGreekIIrules: new	342

<code>\glxtrMathItalicLowerGreekIrules:</code>	<code>\glxtrUpTau:</code>	new	347
new	<code>\glxtrUpTheta:</code>	new	346
<code>\glxtrMathItalicMu:</code>	<code>\glxtrUpUpsilon:</code>	new	347
new	<code>\glxtrUpXi:</code>	new	347
<code>\glxtrMathItalicNabla:</code>	<code>\glxtrUpZeta:</code>	new	346
new	<code>\Iota:</code>	new	322
<code>\glxtrMathItalicNu:</code>	<code>\Kappa:</code>	new	322
new	<code>\Mu:</code>	new	322
<code>\glxtrMathItalicOmega:</code>	<code>\Nu:</code>	new	322
new	<code>\Omicron:</code>	new	322
<code>\glxtrMathItalicOmicron:</code>	<code>\omicron:</code>	new	322
new	<code>\Rho:</code>	new	322
<code>\glxtrMathItalicPartial:</code>	<code>\Tau:</code>	new	322
new	<code>\Upalpha:</code>	new	322
<code>\glxtrMathItalicPhi:</code>	<code>\Upbeta:</code>	new	323
new	<code>\Upchi:</code>	new	323
<code>\glxtrMathItalicPi:</code>	<code>\Upsilon:</code>	new	323
new	<code>\Upeta:</code>	new	323
<code>\glxtrMathItalicPsi:</code>	<code>\Upiota:</code>	new	323
new	<code>\Upkappa:</code>	new	323
<code>\glxtrMathItalicRho:</code>	<code>\Upmu:</code>	new	323
new	<code>\Upnu:</code>	new	323
<code>\glxtrMathItalicSigma:</code>	<code>\Upomicron:</code>	new	323
new	<code>\upomicron:</code>	new	323
<code>\glxtrMathItalicTau:</code>	<code>\Uprho:</code>	new	323
new	<code>\Uptau:</code>	new	323
<code>\glxtrMathItalicTheta:</code>	<code>\Upzeta:</code>	new	323
new	<code>\Zeta:</code>	new	322
<code>\glxtrMathItalicUpperGreekIrules:</code>			
new			
<code>\glxtrMathItalicUpperGreekIrules:</code>			
new			
<code>\glxtrMathItalicUpsilon:</code>			
new			
<code>\glxtrMathItalicXi:</code>			
new			
<code>\glxtrMathItalicZeta:</code>			
new			
<code>\glxtrMathUpGreekIrules:</code>			
new			
<code>\glxtrMathUpGreekIrules:</code>			
new			
<code>\glxtrnonprintablerules:</code>			
new			
<code>\glxtrprovidecommand:</code>			
new			
<code>\glxtrspacerules:</code>			
new			
<code>\glxtrSubScriptDigitrules:</code>			
new			
<code>\glxtrSuperScriptDigitrules:</code>			
new			
<code>\glxtrUpAlpha:</code>			
new			
<code>\glxtrUpBeta:</code>			
new			
<code>\glxtrUpChi:</code>			
new			
<code>\glxtrUpDelta:</code>			
new			
<code>\glxtrUpDigamma:</code>			
new			
<code>\glxtrUpEpsilon:</code>			
new			
<code>\glxtrUpEta:</code>			
new			
<code>\glxtrUpGamma:</code>			
new			
<code>\glxtrUpIota:</code>			
new			
<code>\glxtrUpKappa:</code>			
new			
<code>\glxtrUpLambda:</code>			
new			
<code>\glxtrUpMu:</code>			
new			
<code>\glxtrUpNu:</code>			
new			
<code>\glxtrUpOmega:</code>			
new			
<code>\glxtrUpOmicron:</code>			
new			
<code>\glxtrUpPhi:</code>			
new			
<code>\glxtrUpPi:</code>			
new			
<code>\glxtrUpPsi:</code>			
new			
<code>\glxtrUpRho:</code>			
new			
<code>\glxtrUpSigma:</code>			
new			

1.28 (2018-03-06)	
<code>\@glxtr@docdefval:</code>	changed from count register to macro 14
<code>\@glxtr@dialecthook:</code>	save and restore <code>\TrackLangRequireDialectPrefix</code> 353
<code>\glxtr@redeffield:</code>	changed <code>\csedef</code> to <code>\protected@csedef</code> 33
<code>\glxtr@localsetgrouptitle:</code>	changed <code>\csedef</code> <code>\protected@csedef</code> 119
<code>\glxtr@rsetgrouptitle:</code>	changed <code>\csxdef</code> <code>\protected@csxdef</code> 119
1.29 (2018-04-09)	
<code>\@glxtr@removespaces:</code>	added expansion 122
<code>\@glxtr@dorecord:</code>	don't suppress expansion of <code>\@glxtr@recordloc</code> if counter isn't page 11
<code>\@glxtr@wrglossary@locationhyperlink:</code>	new 23
<code>\glxtr@inc@wrglossaryctr:</code>	new 22
<code>\glxtr@wrglossarylocation:</code>	new 320

\GlsXtrBibTeXEntryAliases: new ..	320	\glsaddpostsetkeys: new	63
\glsxtrfieldforlistloop: corrected		\glsaddpresetkeys: new	63
argument order in \forlistcsloop	31	\glsuserdescription: new	274
\GlsXtrIndexCounterLink: new	320	\glsxtrabbreviationfont: new	57
\GlsXtrInternalLocationHyperlink:		\GlsXtrDualBackLink: new	320
new	23	\GlsXtrDualField: new	320
\GlsXtrProvideBibTeXFields: new ..	321	\GlsXtrExpandedFmt: new	60
indexcounter: new	23	\GLSxtrlong: added \@glsxtr@record	196
\setentrycounter: new	122	\Glsxtrlong: added \@glsxtr@record	195
1.30 (2018-04-25)		\glsxtrlong: added \@glsxtr@record	195
\@glsxtr@record: added check for		\GLSxtrlongpl: added	
post-key hook	9	\@glsxtr@record	200
added check for pre-key hook	9	\Glsxtrlongpl: added	
\@GLSxtr@fullpl: added		\@glsxtr@record	199
\@glsxtr@record	192	\glsxtrlongpl: added	
\@GlsXtrStopUnsetBuffering: new ..	94	\@glsxtr@record	198
\@Glsxtr@fullpl: added		\GLSxtrshort: added	
\@glsxtr@record	192	\@glsxtr@record	194
\@glsxtr@dorecord: don't suppress		\Glsxtrshort: added	
expansion of \@glsrecordlocref ..	11	\@glsxtr@record	193
\@glsxtr@full: added		\glsxtrshort: added	
\@glsxtr@record	189	\@glsxtr@record	193
\@glsxtr@fullpl: added		\GLSxtrshortpl: added	
\@glsxtr@record	191	\@glsxtr@record	198
\@glsxtr@glossadd@postkeys: new ..	10	\Glsxtrshortpl: added	
\@glsxtr@glossadd@prekeys: new ...	10	\@glsxtr@record	197
\@glsxtr@glslink@postkeys: new ...	10	\glsxtrshortpl: added	
\@glsxtr@glslink@prekeys: new	10	\@glsxtr@record	196
\@glsxtr@local@textformat: new ...	60	\GlsXtrStartUnsetBuffering: new ..	93
\@glsxtr@unset: new	93	\GlsXtrStopUnsetBuffering: new ...	93
\@glsxtrbuffer@unset: new	93	indexcounter: added check for	
\glsadd: added \glsaddpostsetkeys ..	63	wrglossary counter	23
added \glsaddpresetkeys	63	\s@GlsXtrStopUnsetBuffering: new ..	94

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	
\.	16, 17, 180, 367
\@	47, 127
\@cGLS@	96, 105
\@cGLSpl@	96, 105
\@cGls@	96, 104
\@cGlspl@	96, 105
\@cgls@	96, 104
\@cglspl@	96, 104
\@do@wrglossary	8, 10, 111
\@do@wrglossary	12–14, 27, 63, 79
\@glo@assign@sortkey	116
\@glo@list	6
\@glo@type	133
\@glossarysec	391
\@gls@expand@field	29
\@glslocalreset	94
\@glslocalunset	94
\@glsreset	94
\@glsunset	93
\@glsxtr@autoindex@escspch	174, 175
\@glsxtr@checkspch	172, 173, 175
\@glsxtr@disabledflycommand	52
\@glsxtr@org@postdescription	115
\@glsxtr@record	14
\@glsxtr@recordcounter	13, 14, 130
\@glsxtrfmt	29
\@glsxtrp	88, 89
\@glsxtrpostloctag	54
\@glsxtrpreloctag	54, 55
\@glsxtrwrglossmark	8, 9, 12, 25, 27, 43, 47, 111
\@newglossaryentry@defcounters	95
\@newglossaryentry@defunitcounters	103
\@par	372
\@ACRlong	85
\@ACRlongpl	85
\@ACRshort	84
\@ACRshortpl	85
\@ACrlong	85
\@ACrlongpl	85
\@ACrshort	84
\@ACrshortpl	85
\@GLS@	84, 99, 100, 144
\@GLSdesc@	68
\@GLSpl@	84, 99, 100, 144
\@GLSplural@	85
\@GLSSymbol@	69
\@GLStext@	85
\@GLSxtr@full	190
\@GLSxtr@fullpl	192
\@GLSxtr@p@acrlong@	85
\@GLSxtr@p@acrlongpl@	85
\@GLSxtr@p@acrshort@	84
\@GLSxtr@p@acrshortpl@	85
\@GLSxtr@p@long@	84
\@GLSxtr@p@longpl@	84
\@GLSxtr@p@plural@	84
\@GLSxtr@p@short@	84
\@GLSxtr@p@shortpl@	84
\@GLSxtr@p@text@	84
\@GLSxtrlong	84, 196
\@GLSxtrlongpl	84, 199
\@GLSxtrp	92
\@GLSxtrshort	84, 194
\@GLSxtrshortpl	84, 198
\@Gls@	84, 98, 99, 143
\@Gls@acentryname	107
\@Gls@entry@field	76, 91, 170
\@Gls@entryname	107
\@GlsXtrEnableOnTheFly	49
\@GlsXtrStopUnsetBuffering	93
\@Glspl@	84, 98, 99, 143
\@Glsplural@	85

<code>\@Gls@text@</code>	85	<code>\@end@glxtr@usesee</code>	41
<code>\@Glsxtr</code>	50, 52	<code>\@end@glxtrifhyphenstart</code>	282, 283
<code>\@Glsxtr@full</code>	190	<code>\@endfortrue</code>	32, 169, 203
<code>\@Glsxtr@fullpl</code>	191	<code>\@firstofone</code>	64, 133, 164, 171, 177
<code>\@Glsxtr@p@acrlong@</code>	85	<code>\@firstofthree</code>	58,
<code>\@Glsxtr@p@acrlongpl@</code>	85		64, 72–75, 81, 189, 191, 193, 195, 197, 198
<code>\@Glsxtr@p@acrshort@</code>	84	<code>\@firstoftwo</code> 65–69, 73, 75, 78, 81, 109, 169,	
<code>\@Glsxtr@p@acrshortpl@</code>	85		170, 181, 182, 190–192, 197–200, 300, 301
<code>\@Glsxtr@p@long@</code>	84	<code>\@for</code>	6, 22, 32, 46, 95, 107,
<code>\@Glsxtr@p@longpl@</code>	84		110, 113, 120, 133, 140, 146, 162, 169, 176
<code>\@Glsxtr@p@plural@</code>	84	<code>\@glo@alias</code>	44, 45
<code>\@Glsxtr@p@short@</code>	84	<code>\@glo@assign@sortkey</code>	113
<code>\@Glsxtr@p@shortpl@</code>	84	<code>\@glo@autosee</code>	25
<code>\@Glsxtr@p@text@</code>	84	<code>\@glo@autoseehook</code>	44
<code>\@Glsxtr@long</code>	84, 195	<code>\@glo@category</code>	101
<code>\@Glsxtr@longpl</code>	84, 199	<code>\@glo@check@sortallowed</code>	113
<code>\@Glsxtrp</code>	91	<code>\@glo@counterprefix</code>	10, 11, 122
<code>\@Glsxtrpl</code>	51, 52	<code>\@glo@countunit</code>	101
<code>\@Glsxtrshort</code>	84, 193	<code>\@glo@default@sorttype</code>	113
<code>\@Glsxtrshortpl</code>	84, 197	<code>\@glo@desc</code>	35, 36
<code>\@acrlong</code>	85	<code>\@glo@descplural</code>	35, 36
<code>\@acrlongpl</code>	85	<code>\@glo@group</code>	13
<code>\@acrshort</code>	84	<code>\@glo@label</code>	
<code>\@acrshortpl</code>	85		12, 13, 28, 41, 44–46, 76, 83, 374–381
<code>\@addtoreset</code>	145	<code>\@glo@location</code>	12
<code>\@afterheading</code>		<code>\@glo@loclist</code>	12
	358, 359, 368–371, 384–387, 394	<code>\@glo@name</code>	172
<code>\@alt@glshyp@opt</code>	81	<code>\@glo@no@assign@sortkey</code>	116
<code>\@auxout</code>	11, 12, 47,	<code>\@glo@parent</code>	376, 377
	56, 97, 105, 110, 111, 123, 127, 129, 130, 395	<code>\@glo@see</code>	41, 42, 45, 46
<code>\@bibgls@restoreat</code>	127, 128	<code>\@glo@seealso</code>	44, 45
<code>\@cGLS</code>	99	<code>\@glo@sort</code>	172
<code>\@cGLS@</code>	96, 99, 105	<code>\@glo@sorttype</code>	113, 120
<code>\@cGLSpl</code>	100	<code>\@glo@text</code>	59
<code>\@cGLSpl@</code>	96, 100, 105	<code>\@glo@thislettergrp</code>	136
<code>\@cGls@</code>	96, 104	<code>\@glo@thisvalue</code>	273
<code>\@cGlspl@</code>	96, 105	<code>\@glo@tmp</code>	28, 42, 76
<code>\@cgls@</code>	96, 104	<code>\@glo@type</code>	46, 82, 107, 110,
<code>\@cglspl@</code>	96, 104		113, 114, 116, 119, 120, 123, 126, 127, 133
<code>\@disable@onlypremakeg</code>	111	<code>\@glo@types</code>	160, 161, 374–380
<code>\@do@auxoutstuff</code>	123, 124	<code>\@glossary@default@style</code> .	52, 53, 114, 389
<code>\@do@glsg@getcounterprefix</code>	10, 11	<code>\@glossarystyle</code>	114
<code>\@do@glsssee</code>	44, 45	<code>\@gls@</code>	84, 98, 99, 142
<code>\@do@newglossaryentry</code>	107, 108, 187	<code>\@gls@@link</code>	59
<code>\@do@seeglossary</code>	13, 14, 25, 47, 111	<code>\@gls@ReturnAfterFi</code>	122
<code>\@do@wrglossary</code>	62, 63, 141	<code>\@gls@actualchar</code>	173
<code>\@empty</code>	64, 72–75, 115, 122, 173, 189–200	<code>\@gls@adjustmode</code>	63
<code>\@end@glxtr@addunused</code>	46	<code>\@gls@alt@hyp@opt</code>	81
<code>\@end@glxtr@gettype</code>	113, 116	<code>\@gls@alt@hyp@opt@char</code>	81

<code>\@gls@alt@hyp@opt@keys</code>	81	<code>\@gls@noidx@nosanitizesort</code>	112
<code>\@gls@automake</code>	113	<code>\@gls@noidx@sanitizesort</code>	112
<code>\@gls@between</code>	119, 120	<code>\@gls@noidx@loclist@finalsep</code>	117
<code>\@gls@checkedmidx</code>	172, 173, 175	<code>\@gls@noidx@loclist@prev</code>	117
<code>\@gls@checkmidxchars</code>	43, 172	<code>\@gls@noidx@loclist@sep</code>	117
<code>\@gls@codepage</code>	123	<code>\@gls@noref@warn</code>	111, 120
<code>\@gls@counter</code>	9, 11, 23, 61, 63, 79, 141	<code>\@gls@org@glsnoidx@displayloc</code>	117, 118
<code>\@gls@currentlettergroup</code>	120, 133, 136	<code>\@gls@org@glsseeformat</code>	117, 118
<code>\@gls@declareoption</code>	5	<code>\@gls@preglossaryhook</code>	114, 177
<code>\@gls@default@longpl</code>	185, 186	<code>\@gls@prevlevel</code>	382, 383, 387, 388
<code>\@gls@doautomake</code>	113, 130	<code>\@gls@quotechar</code>	172
<code>\@gls@doautomake@err</code>	130	<code>\@gls@reference</code>	47, 48, 110, 111
<code>\@gls@enablesavenonumberlist</code>	47	<code>\@gls@restoreat</code>	47
<code>\@gls@encapchar</code>	173	<code>\@gls@saveentrycounter</code>	13, 14, 27, 61, 63, 141
<code>\@gls@entry@count</code>	96, 97	<code>\@gls@see@noindex</code>	26, 127, 128
<code>\@gls@entry@field</code>	28, 32, 33, 44, 76, 89–92, 96, 131, 132	<code>\@gls@setdefault@glslink@opts</code>	9, 30, 61, 80
<code>\@gls@entry@unitcount</code>	105	<code>\@gls@setsort</code>	62, 63
<code>\@gls@field@font</code>	64–71	<code>\@gls@setupsort@none</code>	14
<code>\@gls@field@link</code>	64–71, 76, 77	<code>\@gls@short</code>	185, 186
<code>\@gls@getcounterprefix</code>	10, 11	<code>\@gls@shortpl</code>	183, 186
<code>\@gls@getgrouptitle</code>	119, 133	<code>\@gls@sort</code>	136
<code>\@gls@grptitle</code>	82, 120	<code>\@gls@thisval</code>	169
<code>\@gls@hyp@opt</code>	76, 77, 81, 99, 100, 138, 142–144, 189–199	<code>\@gls@tmp</code>	120
<code>\@gls@hyp@opt@cs</code>	81	<code>\@gls@tmpb</code>	175
<code>\@gls@ifinlist</code>	135	<code>\@gls@type</code>	111–113, 203, 374–381
<code>\@gls@increment@currcount</code>	96	<code>\@gls@write@entrycounts</code>	96
<code>\@gls@increment@currunitcount</code>	104	<code>\@gls@write@entryunitcounts</code>	105
<code>\@gls@keymap</code>	12, 13, 28, 41, 44, 76, 129, 169	<code>\@gls@write@entryunitcounts@do</code>	106
<code>\@gls@label</code>	8, 9, 11, 47, 48, 80, 81, 111, 130, 203	<code>\@gls@xref</code>	12, 43
<code>\@gls@levelchar</code>	173	<code>\@gls@abbrv@current@abbreviation</code>	185, 200
<code>\@gls@link</code>	30, 57, 59, 72–75, 190–200	<code>\@gls@sacronymlists</code>	107
<code>\@gls@link@checkfirsthyper</code>	58, 109	<code>\@gls@doifexistsorwarn</code>	15, 165–168, 170
<code>\@gls@link@label</code>	61, 141	<code>\@gls@entry</code>	97, 105, 106
<code>\@gls@link@nocheckfirsthyper</code>	57, 71–75, 189–200	<code>\@gls@link</code>	62, 82–84
<code>\@gls@link@opts</code>	61	<code>\@gls@localunset</code>	94
<code>\@gls@list</code>	119, 120	<code>\@gls@nextpages</code>	114
<code>\@gls@local@increment@currcount</code>	96	<code>\@gls@nonextpages</code>	114
<code>\@gls@local@increment@currunitcount</code>	104	<code>\@gls@numberformat</code>	9, 11, 61, 63, 79, 141, 169, 172
<code>\@gls@location</code>	136, 137	<code>\@gls@order</code>	110
<code>\@gls@loclist</code>	117, 118, 136, 137	<code>\@gls@pl@</code>	84, 98, 99, 142
<code>\@gls@long</code>	185	<code>\@gls@plural@</code>	85
<code>\@gls@longpl</code>	183, 185, 186	<code>\@gls@punc@token</code>	182
<code>\@gls@map</code>	169	<code>\@gls@recordlocref</code>	10, 11
<code>\@gls@nohyperlist</code>	37, 38	<code>\@gls@showtarget</code>	83
<code>\@gls@noidx@do</code>	120	<code>\@gls@style@altlist</code>	357
<code>\@gls@noidx@getgrouptitle</code>	133	<code>\@gls@style@altlistgroup</code>	358
		<code>\@gls@style@altlisthypergroup</code>	359

<code>\@glsstyle@alttree</code>	371	<code>\@glsxtr@autoindex@escencap</code> ...	173, 174
<code>\@glsstyle@alttreegroup</code>	383	<code>\@glsxtr@autoindex@esclevel</code> ...	173, 174
<code>\@glsstyle@alttreehypergroup</code>	383	<code>\@glsxtr@autoindex@escquote</code> ...	173, 174
<code>\@glsstyle@index</code>	367	<code>\@glsxtr@autoindex@level</code>	173, 174
<code>\@glsstyle@indexgroup</code>	368	<code>\@glsxtr@autoindex@setname</code>	172
<code>\@glsstyle@indexhypergroup</code>	368	<code>\@glsxtr@autoindexcrossrefs</code> 14, 15, 41, 44	
<code>\@glsstyle@inline</code>	367	<code>\@glsxtr@autoseeindexfalse</code>	14
<code>\@glsstyle@list</code>	357	<code>\@glsxtr@autoseeindextrue</code>	16
<code>\@glsstyle@listdotted</code>	356	<code>\@glsxtr@bookindex@atendgroup</code> .	392–394
<code>\@glsstyle@listgroup</code>	358	<code>\@glsxtr@bookindex@atsubendgroup</code> ..	393
<code>\@glsstyle@listhypergroup</code>	358	<code>\@glsxtr@bookindex@atsubsubendgroup</code>	393
<code>\@glsstyle@mcolalttree</code>	387	<code>\@glsxtr@bookindex@between</code>	392, 394
<code>\@glsstyle@mcolalttreegroup</code>	387	<code>\@glsxtr@bookindex@sep</code>	392, 393
<code>\@glsstyle@mcolalttreehypergroup</code> ..	388	<code>\@glsxtr@bookindex@subatendgroup</code> ..	
<code>\@glsstyle@mcolalttreespannav</code>	388	392–394
<code>\@glsstyle@mcolindexgroup</code>	384	<code>\@glsxtr@bookindex@subbetween</code> .	392, 393
<code>\@glsstyle@mcolindexhypergroup</code> ...	384	<code>\@glsxtr@bookindex@subsep</code>	392, 393
<code>\@glsstyle@mcolindexspannav</code>	384	<code>\@glsxtr@bookindex@subsubatendgroup</code>	
<code>\@glsstyle@mcoltreegroup</code>	385	392–394
<code>\@glsstyle@mcoltreehypergroup</code>	385	<code>\@glsxtr@bookindex@subsubbetween</code> ..	
<code>\@glsstyle@mcoltreenonamegroup</code>	386	392, 393
<code>\@glsstyle@mcoltreenonamehypergroup</code>	386	<code>\@glsxtr@bookindexgroupskip</code> ...	392, 394
<code>\@glsstyle@mcoltreenonamespannav</code> ..	387	<code>\@glsxtr@cat</code>	95, 107, 140, 176, 177
<code>\@glsstyle@mcoltreespannav</code>	385	<code>\@glsxtr@checkgroup</code>	134
<code>\@glsstyle@tree</code>	369	<code>\@glsxtr@counterrecordhook</code>	11
<code>\@glsstyle@treegroup</code>	369	<code>\@glsxtr@csname</code>	102, 104
<code>\@glsstyle@treehypergroup</code>	370	<code>\@glsxtr@current@style</code>	52, 389
<code>\@glsstyle@treenoname</code>	370	<code>\@glsxtr@currentunitcount</code>	102, 104
<code>\@glsstyle@treenonamegroup</code>	371	<code>\@glsxtr@currunitcount</code>	103, 105
<code>\@glsstyle@treenonamehypergroup</code> ...	371	<code>\@glsxtr@debugnr</code>	24
<code>\@glstarget</code>	83, 84, 116	<code>\@glsxtr@debugval</code>	24
<code>\@glstext@</code>	85	<code>\@glsxtr@declareoption</code> ...	5, 16, 18, 21, 23
<code>\@glunset</code>	93, 94	<code>\@glsxtr@defaultnoglossarywarning</code> ..	21
<code>\@glswidestname</code>	374, 381	<code>\@glsxtr@defaultnumberformat</code>	
<code>\@glsxtr</code>	50, 52	7, 9, 61, 63, 79, 169, 172
<code>\@glsxtr@@do@@wrglossary</code>	111	<code>\@glsxtr@defpostpunc</code>	16, 17, 25
<code>\@glsxtr@abbreviationsdef</code>	18, 26	<code>\@glsxtr@deprecated@abbrstyle</code>	
<code>\@glsxtr@accessdisplay</code>	169–171	232, 233, 235,
<code>\@glsxtr@activate@initialtagging</code> ..		237, 246, 248, 249, 251, 264, 267, 271, 273	
.....	177, 178	<code>\@glsxtr@disabledflycommand</code>	52
<code>\@glsxtr@addunitcounter</code>	101	<code>\@glsxtr@display@loc</code>	120
<code>\@glsxtr@addunused</code>	46	<code>\@glsxtr@do@@wrintex</code>	80
<code>\@glsxtr@addunusedxrefs</code>	46, 47	<code>\@glsxtr@do@gl:disablehyperinlist</code> ..	78
<code>\@glsxtr@attrval</code>	62, 163–168, 170, 172	<code>\@glsxtr@do@inc@linkcount</code>	146
<code>\@glsxtr@autoindex@at</code>	172–174	<code>\@glsxtr@do@record@wrglossary</code>	8, 14
<code>\@glsxtr@autoindex@doextra@esc</code> ...	172	<code>\@glsxtr@do@redef@forglentries</code>	7
<code>\@glsxtr@autoindex@encap</code>	172–174	<code>\@glsxtr@do@style</code>	22, 319
<code>\@glsxtr@autoindex@esc</code>	172–175	<code>\@glsxtr@do@titlecaps@warn</code>	
<code>\@glsxtr@autoindex@escat</code>	173, 174	164–167, 170, 177

\@glxstr@doabbreviationsdef	18	\@glxstr@longnewglossaryentry	35
\@glxstr@doaccsupp	21, 24	\@glxstr@mark@wordseps	184
\@glxstr@docdefsetting	15	\@glxstr@mark@wordseps@next	184
\@glxstr@docdefval	15, 47, 48	\@glxstr@markwordseps	185, 186
\@glxstr@doccounterrecord	11	\@glxstr@mixed@assign@sortkey	113
\@glxstr@doglossary	133, 134	\@glxstr@noidx@displaynumberlist ..	112
\@glxstr@doiflabelinlist	135	\@glxstr@noidx@do	135
\@glxstr@doloctag	55	\@glxstr@noidx@entrynumberlist	112
\@glxstr@dorecord	8, 10	\@glxstr@noidx@numberlistloop	112
\@glxstr@dorecordnodefer	8, 10	\@glxstr@nomissingglstextnr	21
\@glxstr@dosee@alsoindex@glossary ..	14	\@glxstr@nomissingglstextval	21
\@glxstr@doseeglossary	13, 25	\@glxstr@noop@recordcounter	11, 13
\@glxstr@dostylewarn	203	\@glxstr@nopostpunc	114
\@glxstr@enabletagging	176	\@glxstr@nopostpunc@postdesc	115
\@glxstr@end@	49	\@glxstr@notfoundinlist	182
\@glxstr@endescspch	173–175	\@glxstr@op@recordcounter	14
\@glxstr@entrycount@org@localreset ..	96	\@glxstr@optlist	51
\@glxstr@entrycount@org@localunset ..	96	\@glxstr@org@@starttoc	300
\@glxstr@entrycount@org@reset	96	\@glxstr@org@GLS@	58
\@glxstr@entrycount@org@unset	96	\@glxstr@org@GLSpl@	58
\@glxstr@entryunitcount@org@localreset		\@glxstr@org@Gls@	58
.....	104	\@glxstr@org@Glspl@	58
\@glxstr@entryunitcount@org@localunset		\@glxstr@org@Glsxtrtitlefirst ..	301, 302
.....	104	\@glxstr@org@Glsxtrtitlefirstplural	
\@glxstr@entryunitcount@org@reset ..	104	301, 302
\@glxstr@entryunitcount@org@unset ..	104	\@glxstr@org@Glsxtrtitlefull ..	301, 302
\@glxstr@err@undefaction	7, 13	\@glxstr@org@Glsxtrtitlefullpl	301, 302
\@glxstr@field@linkdefs	57	\@glxstr@org@Glsxtrtitlelong ..	301, 302
\@glxstr@format@overridefalse	171	\@glxstr@org@Glsxtrtitlelongpl	301, 302
\@glxstr@format@overridefalse	171	\@glxstr@org@Glsxtrtitlename ..	301, 302
\@glxstr@foundinlist	182	\@glxstr@org@Glsxtrtitleplural	301, 302
\@glxstr@full	189	\@glxstr@org@Glsxtrtitleshort ..	301, 302
\@glxstr@fullpl	191	\@glxstr@org@Glsxtrtitleshortpl	301, 302
\@glxstr@gettype	113	\@glxstr@org@Glsxtrtitletext ..	301, 302
\@glxstr@glossdescfont	163–165	\@glxstr@org@MakeUppercase	301, 302
\@glxstr@glossnamefont ..	165–168, 170, 171	\@glxstr@org@checkfirsthyper ...	77, 109
\@glxstr@gobbleto@endescspch	175	\@glxstr@org@delimN	55
\@glxstr@groupheading	134, 136	\@glxstr@org@delimR	55
\@glxstr@idx@displaynumberlist	112	\@glxstr@org@doseeglossary	25, 111
\@glxstr@idx@entrynumberlist	112	\@glxstr@org@gloautoosee	26
\@glxstr@ifcsstart	49	\@glxstr@org@gls@	57, 58
\@glxstr@ifpunctoken	182	\@glxstr@org@glsdohypertarget	116
\@glxstr@ifunitcounter	101	\@glxstr@org@glsignore	55
\@glxstr@insert@dots	184	\@glxstr@org@glspl@	58
\@glxstr@insert@dots@next	184	\@glxstr@org@glsxtrtitlefirst ..	301, 302
\@glxstr@insert@dots	185	\@glxstr@org@glsxtrtitlefirstplural	
\@glxstr@label	32, 46, 47, 146, 162, 163	301, 302
\@glxstr@loadstyles	355, 356	\@glxstr@org@glsxtrtitlefull ..	301, 302
\@glxstr@local@textformat	61, 62	\@glxstr@org@glsxtrtitlefullpl	301, 302

\@glxtr@org@glxtrtitlelong ..	301, 302	\@glxtr@recordsee	14, 25, 43
\@glxtr@org@glxtrtitlelongpl	301, 302	\@glxtr@redef@for@gl@entries	7, 26
\@glxtr@org@glxtrtitlename ..	301, 302	\@glxtr@redefstyles	21, 22, 319
\@glxtr@org@glxtrtitleorpdforheading	301, 302	\@glxtr@reg@glosslist	110–113, 116
\@glxtr@org@glxtrtitleplural	301, 302	\@glxtr@restore@postpunc	115
\@glxtr@org@glxtrtitleshort ..	301, 302	\@glxtr@rglstrigger@record ...	142–144
\@glxtr@org@glxtrtitleshortpl	301, 302	\@glxtr@s@longnewglossaryentry	35
\@glxtr@org@glxtrtitletext ..	301, 302	\@glxtr@savepreloctag	54, 56
\@glxtr@org@makeglossaries	110	\@glxtr@setentrycountunsetattr	95
\@glxtr@org@markboth	300	\@glxtr@setentryunitcountunsetattr	106
\@glxtr@org@markright	299, 300	\@glxtr@setupshortcuts	20, 21, 26
\@glxtr@org@newacronymstyle ..	108, 109	\@glxtr@shortcutsnr	20
\@glxtr@org@postdescription ..	115, 178	\@glxtr@shortcutsval	20, 129
\@glxtr@org@see@noindex	127, 128	\@glxtr@swaptwo	182
\@glxtr@org@setacronymstyle ..	108, 109	\@glxtr@tag	177
\@glxtr@org@theHvalue	8, 9	\@glxtr@taggingcs	177
\@glxtr@org@unset@buffer	93, 94	\@glxtr@textformat	62
\@glxtr@org@prefix	10, 11	\@glxtr@theHvalue ...	8–10, 60, 61, 63, 141
\@glxtr@org@printglossary	52, 115	\@glxtr@thevalue	8–10, 60, 61, 63, 141
\@glxtr@org@warndep	183	\@glxtr@thisloctag	55, 56
\@glxtr@p@acrlong@	85	\@glxtr@titlelabel	118, 119, 135
\@glxtr@p@acrlongpl@	85	\@glxtr@tmp	22, 60, 121
\@glxtr@p@acrshort@	84	\@glxtr@type	163
\@glxtr@p@acrshortpl@	85	\@glxtr@unitcountlist	101
\@glxtr@p@long@	84	\@glxtr@unset	93, 94
\@glxtr@p@longpl@	84	\@glxtr@unset@buffer	93, 94
\@glxtr@p@plural@	84	\@glxtr@unsrt@getgroup@title	133
\@glxtr@p@short@	84	\@glxtr@usesee	41
\@glxtr@p@shortpl@	84	\@glxtr@warn@onexistsordo	7, 14
\@glxtr@p@text@	84	\@glxtr@warn@undefaction	7, 14
\@glxtr@pagetag	54, 55	\@glxtr@wrglossary@locationhyperlink	24
\@glxtr@pagetag	54, 55	\@glxtr@wrglossnr	59
\@glxtr@prevunitcount	103	\@glxtr@wrglossval	59
\@glxtr@printglossnr	115, 116	\@glxtr@buffer@unset	93
\@glxtr@printglossopts	52, 113, 115	\@glxtr@dialecthook	319
\@glxtr@printglossval	115	\@glxtr@docdef@false	48
\@glxtr@printunsrtglossaryskipentry	133, 134	\@glxtr@entryfmt	30
\@glxtr@provide@addstoragekey	29	\@glxtr@fmt	29
\@glxtr@provide@storagekey	28	\@glxtr@glossentry	131
\@glxtr@record	13, 14, 57–59, 63, 189, 191–200	\@glxtr@glossentry@other	131, 132
\@glxtr@record@setting	8, 10, 13, 43, 47, 48, 110	\@glxtr@hypername@prefix	116, 135
\@glxtr@record@setting@alsoindex ..	8, 10, 43, 110	\@glxtr@ifhasfield	32
\@glxtr@record@setting@off	47	\@glxtr@ifhyphenstart	282
\@glxtr@record@setting@only	110	\@glxtr@index@aliased	79
		\@glxtr@index@crossrefs@false	15
		\@glxtr@index@crossref@true	15
		\@glxtr@inmark	300
		\@glxtr@long	84, 194, 195

short-hyphen-postlong-hyphen . . .	293, 296	\ACSP	19
short-long-user	277	\Acsp	19
short-nolong	216, 218	\acsp	19
short-nolong-desc	218	\actualchar	175
short-postlong-user	279	\addtolength	382
\abbreviationsname	17	\advance	97, 106, 128, 138
\abbrvpluralsuffix		\AF	18
.	129, 186, 207, 209, 212, 214,	\Af	18
.	215, 217, 220, 224, 225, 227, 228, 231,	\af	18
.	232, 234, 236, 238, 240, 241, 243, 245,	\AFP	18
.	246, 248, 250, 252, 254, 256, 257, 259,	\Afp	18
.	260, 262, 264, 266, 267, 270, 272, 274,	\afp	18
.	276, 278, 281, 284, 286, 289, 292, 294, 297	\AL	18
\ABP	18	\Al	18
\Abp	18	\al	18
\abp	18	\ALP	18
\AC	19	\Alp	18
\Ac	19	\alp	18
\ac	18	amsmath package	23
\ACF	19	\AnyTrackedLanguages	319, 353
\Acf	19	\appto	12, 13, 22, 28, 41, 43–45,
\acf	19	76, 80, 95, 103, 134, 135, 171, 181, 184, 355
\ACFP	19	\arabic	23, 394
\Acfp	19	\AS	18
\acfp	19	\As	18
\ACL	19	\as	18
\Acl	19	\ASP	18
\acl	19	\Asp	18
\ACLP	19	\asp	18
\Aclp	19	\AtBeginDocument	24, 27, 53, 54, 129
\aclp	19	\AtEndDocument	46, 96, 105, 123
\ACP	19		
\Acp	19	B	
\acp	18	babel package	172, 173, 181
\ACRfullfmt	108	\begin	120, 125, 133, 357, 359–366, 385–388, 392
\Acrfullfmt	108	\begingroup	8, 9, 29, 30, 79, 131, 132, 145
\acrfullfmt	108	\bgroup	35, 36, 114
\ACRfullplfmt	108	bib2gls	23, 30, 136, 141, 142, 319–321
\Acrfullplfmt	108		
\acrfullplfmt	108	C	
\acronymentry	107	\c@wrglossary	23
\acronymfont	72–75, 87, 108, 109	\catcode	47, 127
\acronymname	17	category attributes:	
\acronymsort	107	apospplural	186
\acronymtype	17, 107, 108	discardperiod	180
\acrpluralsuffix	107, 129	entrycount	93–95, 97, 106
\ACS	19	firstuc	167
\Acs	19	glossdesc	163
\acs	18	glossdescfont	163
		glossname	165

\do@gl:disablehyperinlist 61, 79
doc package 175
\dolistcsloop 31
\DTLifinlist 111, 112, 116
\DTLifint 118

E

\eappto 11, 22, 36–39, 134, 136, 172, 355
\edef 6, 8–11, 36–39, 42, 44,
45, 47, 61–63, 78, 79, 82, 83, 101, 102,
104, 110, 111, 116, 118, 121–123, 127,
131, 132, 141, 145, 163–166, 168–170,
172, 173, 175, 183, 353, 376, 377, 392, 393
\eglssetwidest 374–381
\egroup 35, 36, 114
\else 8–12, 15–17,
20, 21, 25, 30, 34, 48, 49, 54, 56, 59, 62,
79, 80, 97, 109, 110, 112, 114–116, 118,
121, 122, 125, 127, 128, 141, 171–173,
175, 182, 184, 186, 193–200, 202, 203,
207, 209, 210, 212–221, 224, 226–254,
256–272, 275, 276, 278, 279, 281–283,
285, 288, 290, 291, 294, 295, 297, 298,
357, 359–369, 371, 382, 383, 389, 391, 393
\emph 251, 252
\empty 120–122
\encapchar 175
\end 120,
125, 126, 134, 357, 359–366, 385–388, 392
\end@gl:extr@display@loc 120, 121
\endcsname 6, 29,
37, 38, 43, 47, 52, 56, 59, 61, 63, 72–77,
79, 88, 102, 111, 119, 123, 126, 127, 133,
138–141, 146, 163, 183, 190–200, 206, 381
\endgroup 8, 10, 30, 79, 131–133, 145
\ensuremath 24
entry categories:
 abbreviation 200
 general 157, 159
 index 161
\epreto 172
\equal 126, 127
etoolbox package 5
\expandafter 24, 29, 30, 32,
41, 42, 46, 47, 49–51, 60, 76, 77, 81, 88,
100, 101, 111–113, 116, 119, 121, 122,
133, 134, 136, 138, 145, 163, 166, 167,
169, 171, 172, 175, 182, 185, 186, 282, 392
\expandonce 107, 136, 173, 208, 286

F

\fi .. 7–12, 14–16, 18, 20, 21, 24–26, 30, 34,
41, 43, 44, 46–49, 53, 54, 56, 59, 60, 62,
63, 79, 80, 97, 105, 106, 109, 112–116,
118, 121, 122, 124–128, 130, 141, 171–
173, 175, 182, 184, 186, 193–200, 202,
203, 207, 209, 210, 212–221, 224, 226–
254, 256–272, 274–276, 278, 279, 281–
283, 285, 288, 290, 291, 294, 295, 297,
298, 357, 359–371, 373–383, 389, 391, 394
first use 396
 flag 396
 text 396
\firstacronymfont 108, 109
fontspec package 129
\footnote 211
\foralllglossaries
..... 46, 133, 161, 163, 374, 375, 377–381
\foralllglsentries 97, 106
\ForEachTrackedDialect 319, 353
\forlglsentries 6, 46, 161, 163, 374–381
\forallistcsloop 31, 106, 120
\forallistloop 94, 117, 177
\futurelet 182

G

\gdef 55, 174
\Genacrffullformat 108
\genacrffullformat 108
\GenericAcronymFields 108
\Genplacrffullformat 108
\genplacrffullformat 108
\glo@grabfirst 136
\glo@name 166, 167, 170, 171
\gloaliaslabel 83
\global 10, 35, 36, 114, 136
\glolinkprefix 62, 83, 129
glossaries package ... 14, 25, 26, 41–45, 113, 356
glossaries-accsupp package 21, 24, 147
glossaries-extra package 2, 353
glossaries-extra-bib2gls package 14, 27, 319, 353
glossaries-extra-stylemods package . 21, 179, 319
glossaries-stylemods package 390
glossaries.sty package 36
\GlossariesExtraWarning 6,
16, 34, 35, 49, 51, 62, 109, 111, 121,
125, 128, 133, 163–166, 168, 170, 177, 205
\GlossariesExtraWarningNoLine 16, 97, 106
\GlossariesWarning 55, 112, 114, 117, 118, 203

<code>\GlossariesWarningNoLine</code>	111, 124	<code>\glossarytoctitle</code> .	37, 38, 113, 120, 126, 133
<code>glossary</code> styles:		<code>\glossentry</code>	114,
<code>altlist</code>	357		137, 356, 357, 359–366, 368–370, 382, 392
<code>altlistgroup</code>	358	<code>\glossentrydesc</code>	356–366, 368–372
<code>altlisthypergroup</code>	359	<code>\glossentryname</code>	
<code>alttree</code>	371, 372, 382	 131, 356–366, 368–370, 382, 383, 390
<code>alttreegroup</code>	383	<code>\glossentrynameother</code>	132
<code>alttreehypergroup</code>	383	<code>\glossentrysymbol</code>	360–364, 366, 368–370, 372
<code>index</code>	367	<code>\glossxtrsetpopts</code>	178
<code>indexgroup</code>	368	<code>\GLS</code>	95, 106, 140
<code>indexhypergroup</code>	368	<code>\Gls</code>	50, 95, 106, 140
<code>inline</code>	367	<code>\gls</code>	34, 50, 52, 95, 106, 112, 124, 140
<code>list</code>	357	<code>\gls@assign@desc</code>	35, 36
<code>listdotted</code>	356	<code>\gls@assign@field</code>	12, 13, 28, 76
<code>listdottedstyle</code>	357	<code>\gls@checkseeallowed</code>	49, 111
<code>listgroup</code>	358	<code>\gls@codepage</code>	123
<code>listhypergroup</code>	358	<code>\gls@defdocnewglossaryentry</code> ..	47, 95, 103
<code>mcolalttree</code>	387	<code>\gls@defglossaryentry</code>	35, 36, 50, 51
<code>mcolalttreegroup</code>	387	<code>\gls@dotocitle</code>	114
<code>mcolalttreehypergroup</code>	388	<code>\gls@glossary</code>	43
<code>mcolalttreespannav</code>	388	<code>\gls@grplabel</code>	82
<code>mcolindexgroup</code>	384	<code>\gls@level</code>	136, 137
<code>mcolindexhypergroup</code>	384	<code>\gls@noidxglossary</code>	111
<code>mcolindexspannav</code>	384	<code>\gls@org@glossaryentryfield</code>	114
<code>mcoltreegroup</code>	385	<code>\gls@org@glossarysubentryfield</code>	114
<code>mcoltreehypergroup</code>	385	<code>\gls@orgTrackLangRequireDialectPrefix</code>	
<code>mcoltreenonamegroup</code>	386	 353
<code>mcoltreenonamehypergroup</code>	386	<code>\gls@save@numberlist</code>	54, 56
<code>mcoltreenonamespannav</code>	387	<code>\gls@set@xr@key</code>	43, 44
<code>mcoltreespannav</code>	385	<code>\gls@tmplen</code>	374–383
<code>sublistdotted</code>	357	<code>\gls@type</code>	111
<code>tree</code>	369	<code>\glsabbrvdefaultfont</code> ...	189, 207, 209,
<code>treegroup</code>	369		212, 214, 215, 217, 220, 273, 283, 286, 296
<code>treehypergroup</code>	370	<code>\glsabbrvemfont</code>	
<code>treenoname</code>	370	 251–260, 262, 264, 266, 268–272
<code>treenonamegroup</code>	371	<code>\glsabbrvfont</code>	
<code>treenonamehypergroup</code>	371		.. 85, 86, 108, 189, 193, 194, 197, 198,
<code>glossary-bookindex</code> package	356		200, 202, 203, 206–215, 217, 220, 222,
<code>glossary-hypernav</code> package	82		224, 225, 227, 228, 231, 232, 234, 236,
<code>glossary-long</code> package	361		238, 240, 241, 243, 245, 246, 248, 250,
<code>glossary-longbooktabs</code> package	361		252, 254, 256, 257, 259, 260, 262, 264,
<code>\glossaryentrynumbers</code>	56, 114, 137		266, 268, 270, 272, 274, 276, 278, 280–
<code>\glossaryheader</code>	120,		282, 284, 286, 288–290, 292, 294, 295, 297
	133, 357–366, 368–371, 382–386, 388, 392	<code>\glsabbrvhyphenfont</code>	
<code>\glossaryname</code>	113	 283–285, 289, 290, 292–294, 296
<code>\glossarypostamble</code>	120, 134, 135	<code>\glsabbrvonlyfont</code>	296–299
<code>\glossarypreamble</code>	120, 133	<code>\glsabbrvscfont</code> .	223–228, 231, 232, 234–236
<code>\glossarysection</code>	120, 126, 133, 135	<code>\glsabbrvsmfont</code>	
<code>\glossarytitle</code> .	37, 38, 113, 114, 120, 126, 133	 237–241, 243, 245, 246, 248, 250

<code>\glsabbrvuserfont</code>	273–279, 281	240, 242, 243, 249, 251, 256, 258, 259,
<code>\GLSaccessdesc</code>	68	261, 270–272, 278, 279, 281, 290, 295, 298
<code>\Glsaccessdesc</code>	68, 164, 176	<code>\glsaccessshortpl</code>
<code>\glsaccessdesc</code>	67, 164, 180 73, 188, 197, 198, 202, 207,
<code>\GLSaccessdescplural</code>	68	209, 212, 214–219, 221, 224, 226–231,
<code>\Glsaccessdescplural</code>	68	233–240, 242–247, 249–254, 256, 258–
<code>\glsaccessdescplural</code>	68	263, 265–268, 270, 272, 275, 276, 278,
<code>\GLSaccessfirst</code>	65	279, 281, 284, 286, 287, 289, 292, 295, 298
<code>\Glsaccessfirst</code>	65	<code>\GLSaccesssymbol</code>
<code>\glsaccessfirst</code>	65	<code>\Glsaccesssymbol</code>
<code>\GLSaccessfirstplural</code>	67	<code>\glsaccesssymbol</code>
<code>\Glsaccessfirstplural</code>	67	<code>\GLSaccesssymbolplural</code>
<code>\glsaccessfirstplural</code>	66	<code>\Glsaccesssymbolplural</code>
<code>\Glsaccesslong</code>	74, 187,	<code>\glsaccesssymbolplural</code>
195, 207, 216, 220, 221, 224, 230–233,		<code>\GLSaccessstext</code>
238, 244–247, 252, 254, 262–266, 268,		<code>\Glsaccessstext</code>
275, 276, 284, 286, 287, 290, 292, 297, 298		<code>\glsaccessstext</code>
<code>\glsaccesslong</code>		<code>\glsacrshortcutstrue</code>
..... 74, 187, 195, 196, 207, 209, 210,		<code>\glsacspacemax</code>
212, 214, 215, 217, 219–221, 224, 226,		<code>\glsadd</code>
227, 229–235, 237, 238, 240–249, 251,		30, 47, 124
252, 254, 256, 258–272, 274, 276, 278,		<code>\glsadd options</code>
279, 281, 284, 286, 287, 290, 292, 297, 298		<code>theHvalue</code>
<code>\Glsaccesslongpl</code>	75,	<code>thevalue</code>
188, 199, 207, 216, 220, 221, 224, 230,		9, 10
231, 233, 238, 244–247, 253, 254, 262–		<code>\glsaddpostsetkeys</code>
268, 275, 276, 284, 286, 287, 290, 292, 298		10, 63
<code>\glsaccesslongpl</code> .. 75, 188, 199, 200, 207,		<code>\glsaddpresetkeys</code>
209, 210, 212–215, 217–221, 224, 226–		10, 63
235, 237, 238, 240, 242–249, 251, 252,		<code>\glsaddstoragekey</code>
254, 256, 258, 259, 261–273, 275, 276,		45, 157, 321
279, 281, 284, 286, 287, 290, 292, 297, 298		<code>\glsbackslash</code>
<code>\GLSaccessname</code>	67	49
<code>\Glsaccessname</code>	67	<code>\glscapscase</code>
<code>\glsaccessname</code>	42, 67	58, 64–75, 77, 189–202
<code>\GLSaccessplural</code>	66	<code>\glscategory</code> .. 56, 64, 78, 85, 86, 158–160,
<code>\Glsaccessplural</code>	66	163–170, 176, 179, 180, 189–194, 196–198
<code>\glsaccessplural</code>	66	<code>\glscategorylabel</code>
<code>\Glsaccessshort</code>	72, 194, 203, 209, 78, 183, 185, 186, 213, 235,
212, 214, 217, 219, 226, 227, 229, 234–		250, 271, 275, 277–279, 289, 291, 294, 296
237, 240, 242, 243, 249–251, 256, 258,		<code>\glsclsebrace</code>
259, 261, 270–272, 278, 279, 281, 289, 295		43, 125–127
<code>\glsaccessshort</code> .. 72, 187, 193, 194, 202,		<code>\glscounter</code>
207, 209, 212, 214–219, 221, 224, 226–		9
231, 233–238, 240–252, 254, 256, 257,		<code>\glscurrententrylabel</code>
259–263, 265–268, 270, 272, 275, 276,		.. 54–56, 114, 122, 123, 131–134, 178, 179
278, 281, 284, 286, 287, 289, 292, 295, 298		<code>\glscurrentfieldvalue</code> .. 30, 32, 34, 273, 320
<code>\Glsaccessshortpl</code> 73, 197, 203, 210, 212–		<code>\glscustomtext</code> .. 57, 58, 72–75, 189–200, 202
214, 218, 219, 226, 228, 229, 235–237,		<code>\glsdefaulttype</code>
	 6, 17, 34, 113, 124, 132, 133, 135
		<code>\glsdescriptionaccessdisplay</code> .. 151, 164
		<code>\glsdescriptionpluralaccessdisplay</code>
	 151, 152
		<code>\glsdescwidth</code>
		359–366
		<code>\glsdetoklabel</code>
		8, 9, 31–36,
		39–42, 46–49, 61, 63, 79, 83, 96, 101–
		106, 111, 114, 117, 118, 131, 132, 136,
		139–141, 163, 166, 167, 170, 171, 376, 377

<code>\glsdisplaynumberlist</code>	112, 116	<code>\glsentryshort</code>	85–
<code>\glsdohyperlink</code>	81, 82, 84		87, 109, 152, 156, 275, 277, 288, 312, 313
<code>\glsdohypertarget</code>	84, 116	<code>\Glsentryshortpl</code>	86, 87, 153, 157
<code>\glsdoifexists</code>		<code>\glsentryshortpl</code>	
.....	15, 25, 33, 39, 41–43, 57, 58, 63,		86, 87, 153, 156, 157, 312, 313
	71–75, 83, 111, 117, 118, 131, 132, 189–200	<code>\Glsentrysymbol</code>	150, 155
<code>\glsdoifexistsordo</code>	29, 30, 59	<code>\glsentrysymbol</code>	150, 155, 378–380
<code>\glsdoifexistsorwarn</code>	15, 163, 164, 176	<code>\Glsentrysymbolplural</code>	151, 156
<code>\glsdoifnoexists</code>	35	<code>\glsentrysymbolplural</code>	150, 151, 156
<code>\glsdonohyperlink</code>	62, 83, 84	<code>\Glsentrytext</code>	148, 154
<code>\glsdosanitizesort</code>	112	<code>\glsentrytext</code>	83, 147, 148, 154, 155, 314
<code>\glsenableentrycount</code>	94, 97, 105	<code>\glsentrytype</code>	131, 132
<code>\glsenableentryunitcount</code>	96, 106	<code>\Glsentryuseri</code>	70
<code>\glsentrycounter</code>	23, 122	<code>\glsentryuseri</code>	70
<code>\glsentrycurrcount</code>	95, 97, 103	<code>\Glsentryuserii</code>	70
<code>\Glsentrydesc</code>	151, 156, 164	<code>\glsentryuserii</code>	70
<code>\glsentrydesc</code>	151, 156, 165	<code>\Glsentryuseriii</code>	70
<code>\Glsentrydescplural</code>	152, 156	<code>\glsentryuseriii</code>	70
<code>\glsentrydescplural</code>	151, 152, 156	<code>\Glsentryuseriv</code>	70
<code>\Glsentryfirst</code>	100, 144, 149, 155	<code>\glsentryuseriv</code>	71
<code>\glsentryfirst</code>	100, 144, 149, 155, 315	<code>\Glsentryuserv</code>	71
<code>\Glsentryfirstplural</code> ...	101, 145, 149, 155	<code>\glsentryuserv</code>	71
<code>\glsentryfirstplural</code>		<code>\Glsentryuservi</code>	71
.....	100, 144, 149, 150, 155, 315, 316	<code>\glsentryuservi</code>	71
<code>\glsentryfmt</code>	36–39	<code>\glsextrapostnamehook</code>	169
<code>\Glsentryfull</code>	108	<code>\glsfieldfetch</code>	83
<code>\glsentryfull</code>	108	<code>\glsfieldxdef</code>	162, 163
<code>\Glsentryfullpl</code>	108	<code>\glsfindwidesttoplevelname</code>	374
<code>\glsentryfullpl</code>	108	<code>\GLSfirst</code>	307
<code>\glsentryitem</code>	131,	<code>\Glsfirst</code>	307
	132, 356, 357, 359–366, 368–370, 382, 393	<code>\glsfirst</code>	307
<code>\Glsentrylong</code>	86, 87, 100, 144, 153, 157	<code>\glsfirstabbrvdefaultfont</code>	
<code>\glsentrylong</code>	86, 87, 100, 144, 153,		188, 207, 209, 212, 214, 215, 217, 220, 286
	157, 213, 236, 250, 271, 278, 279, 294, 316	<code>\glsfirstabbrvfont</code>	252–272
<code>\Glsentrylongpl</code>	86, 87, 101, 154, 157	<code>\glsfirstabbrvfont</code>	108, 187, 188,
<code>\glsentrylongpl</code> ...	86, 87, 100, 154, 157, 317		207–221, 224, 225, 227, 229, 231, 232,
<code>\Glsentrylongplural</code>	145		234, 236, 238, 240, 241, 243, 245, 246,
<code>\glsentrylongplural</code>	144		248, 250, 252, 254, 256, 257, 259, 260,
<code>\Glsentryname</code>	147, 154, 165, 167, 168		262, 264, 266, 268, 270, 272, 274, 276,
<code>\glsentryname</code>			278, 281, 284, 286, 287, 289, 292, 294, 297
.....	131, 147, 154, 172, 313, 374–381, 395	<code>\glsfirstabbrvhyphenfont</code>	
<code>\glsentrynumberlist</code>	112, 118, 379–381		283–285, 288, 289, 291–296
<code>\Glsentryplural</code>	148, 155	<code>\glsfirstabbrvonlyfont</code>	297, 298
<code>\glsentryplural</code>	148, 155, 314, 315	<code>\glsfirstabbrvscfont</code>	223–237
<code>\glsentryprevcount</code>	96, 97, 103	<code>\glsfirstabbrvsmfont</code>	238–251
<code>\glsentryprevmaxcount</code>	104	<code>\glsfirstabbrvuserfont</code>	274–282
<code>\glsentryprevtotalcount</code>	103	<code>\glsfirstaccessdisplay</code>	149
<code>\Glsentryshort</code>	85, 87, 152, 156		

<code>\glsfirstlongdefaultfont</code>	<code>\glsifplural</code>
207, 209, 215, 217, 220, 223–233, 238–	.. 58, 64, 66–69, 71–75, 180, 181, 189–201
248, 252, 253, 255, 256, 259–263, 266, 267	<code>\glsifregular</code> 56, 64, 100, 101, 144, 145
<code>\glsfirstlongemfont</code>	<code>\glsifregularcategory</code> 160
.... 253–255, 257, 258, 264, 265, 267–269	<code>\glsifusetranslator</code> 37
<code>\glsfirstlongfont</code> ... 187, 188, 207–210,	<code>\glsignore</code> 55
212, 214–222, 224, 226, 227, 229, 231,	<code>\glsinlinedescformat</code> 367
232, 234, 236, 238, 240, 241, 243, 245,	<code>\glsinlinesubdescformat</code> 367
246, 248, 250, 252, 254, 256, 257, 259,	<code>\glsinsert</code> 59,
260, 262, 264, 266, 268, 270, 272, 274,	64, 72–75, 189–202, 282, 289, 291, 294, 296
276, 278, 281, 284, 286, 289, 292, 294, 297	<code>\glskeylisttok</code> 107, 108, 185, 187
<code>\glsfirstlongfootnotefont</code>	<code>\glslabel</code> 8, 9, 29, 39, 42,
.... 211–214, 234–237, 248–251, 269–273	56, 57, 59–62, 78, 79, 82, 83, 109, 141,
<code>\glsfirstlonghyphenfont</code> 283–294	145, 146, 179, 180, 200–202, 213, 236,
<code>\glsfirstlongonlyfont</code> 297, 298	250, 271, 275, 277–279, 289, 291, 294, 296
<code>\glsfirstlonguserfont</code> 274–282	<code>\glslabeltok</code> . 107, 185, 187, 207–211, 213,
<code>\GLSfirstplural</code> 308	215–220, 222–228, 230, 234, 236, 238–
<code>\Glsfirstplural</code> 308	241, 243, 245, 248, 250, 252–260, 262,
<code>\glsfirstplural</code> 308	264, 265, 267, 269–271, 274, 275, 277–
<code>\glsfirstpluralaccessdisplay</code> .. 149, 150	282, 284–287, 289, 291–294, 296, 297, 299
<code>\glsforeachincategory</code> 203	<code>\glsletentryfield</code> 172
<code>\glsgenentryfmt</code> 56, 57	<code>\glslink</code> 108
<code>\glsgetattribute</code> 62, 82, 97, 101–	<code>\glslink options</code>
103, 123, 141, 145, 163–166, 168, 170, 172	counter 10
<code>\glsgetcategoryattribute</code> 158	format 171
<code>\glsgetgrouptitle</code> 358, 359, 368–371, 383–389	hyper 299
<code>\glsgetwidestname</code> 372	hyperoutside 60
<code>\glsgroupheading</code>	noindex 8, 9, 78, 299
.... 136, 357–366, 368–371, 382–388, 394	textformat 62
<code>\glsgroupskip</code> 136, 357, 359–369, 371, 383, 394	theHvalue 61
<code>\glshasattribute</code>	thevalue 61, 138
..... 62, 82, 97, 102, 104, 106, 122,	wrgloss 8, 59
141, 145, 163–168, 170, 171, 207–211,	<code>\glslinkcheckfirsthyperhook</code> 78
213, 216, 218, 219, 222, 223, 225, 226,	<code>\glslinkpostsetkeys</code> 10, 61, 141
234, 236, 238–241, 248, 250, 252–255,	<code>\glslinkpresetkeys</code> 10, 61, 141
257, 258, 265, 269–271, 274, 275, 277–	<code>\glslinkvar</code> 81
282, 284–287, 289, 291–294, 296, 297, 299	<code>\glslistchildpostlocation</code> 357
<code>\glshascategoryattribute</code> 159	<code>\glslistchildprelocation</code> 357, 358
<code>\glshex</code> .. 324–329, 332–338, 340–343, 345–353	<code>\glslistdottedwidth</code> 356
<code>\glshyperlink</code> 83, 320	<code>\glslistgroupheaderfmt</code> 358, 359
<code>\glshypernavsep</code> 120	<code>\glslistnavigationitem</code> 358, 359
<code>\glshypernumber</code> 123, 171	<code>\glslistprelocation</code> 357, 358
<code>\glsifattribute</code> 59, 60, 65, 78, 80, 89, 145,	<code>\glslocalunset</code> 59, 141
161, 164–167, 170, 178, 180, 181, 303–311	<code>\glslongaccessdisplay</code> 153
<code>\glsifcategory</code> 161	<code>\glslongdefaultfont</code> ... 189, 207, 209,
<code>\glsifcategoryattribute</code>	211, 215, 217, 220, 224, 226, 227, 229–
..... 78, 159, 160, 185, 186	233, 238, 240, 241, 243, 245–247, 252,
<code>\glsifnotregular</code> 64	256, 259, 260, 262, 263, 266, 273, 283, 296
<code>\glsifnotregularcategory</code> 160	<code>\glslongemfont</code> .. 251, 254, 257, 264, 267, 268

<code>\glslongfont</code>	86, 87, 189, 195, 196, 199, 200, 207–210, 212, 214, 215, 217, 220, 222, 224, 226, 227, 229, 231, 232, 234, 236, 238, 240, 241, 243, 245, 246, 248, 250, 252, 254, 256, 257, 259, 260, 262, 264, 266, 268, 270, 272, 274, 276, 278, 281, 284, 286, 289, 292, 294, 297, 298
<code>\glslongfootnotefont</code>	211, 212, 214, 234, 236, 248, 250, 270, 272
<code>\glslonghyphenfont</code>	283, 284, 286, 287, 289, 292, 294
<code>\glslongonlyfont</code>	296, 297
<code>\glslongpltok</code> 186, 187, 207–211, 220, 222–226, 230, 234, 238, 239, 241, 245, 248, 252–258, 262, 264, 267, 269, 274, 275, 277, 279– 282, 284–287, 289, 290, 292, 293, 297, 298
<code>\glslongpluralaccessdisplay</code>	154
<code>\glslongtok</code> 107, 185, 187, 207–211, 213, 215, 217, 220, 222–228, 230, 234, 235, 238, 239, 241, 243, 245, 248, 250, 252–260, 262, 264, 267, 269, 271, 274, 275, 277, 279– 282, 284–287, 289, 290, 292–294, 297, 298
<code>\glslonguserfont</code>	274, 276–279, 281
<code>\glsncols</code>	385–388
<code>\GLSname</code>	304, 305
<code>\Glsname</code>	305
<code>\glsname</code>	304, 305
<code>\glsnameaccessdisplay</code> ..	147, 165, 166, 168
<code>\glsnamefont</code>	165, 167, 168, 170
<code>\glsnavhyperlink</code>	120
<code>\glsnavhyperlinkname</code>	82
<code>\glsnavhypertarget</code>	358, 359, 369–371, 384–389
<code>\glsnavigation</code> ..	358, 359, 369–371, 383–388
<code>\glsnextpages</code>	114
<code>\glsnoidxdisplayloc</code>	117, 118
<code>\glsnoidxdisplayloclisthandler</code>	117
<code>\glsnoidxloclist</code>	118, 137
<code>\glsnoidxnumberlistloophandler</code>	117
<code>\glsnonextpages</code>	114
<code>\glsnonumberlistfalse</code>	54
<code>\glsnonumberlisttrue</code>	54
<code>\glsnopostdotfalse</code>	115
<code>\glsnopostdottrue</code>	115
<code>\glsnumberlistloop</code>	112
<code>\glsnumlistlastsep</code>	117
<code>\glsnumlistsep</code>	117
<code>\glsopenbrace</code>	43, 125–127
<code>\glsorder</code>	110
<code>\glspagelistwidth</code>	360, 362, 364, 366
<code>\glspar</code>	135
<code>\GLSpl</code>	95, 106, 140
<code>\Glspl</code>	51, 95, 106, 140
<code>\glspl</code>	51, 95, 106, 140
<code>\GLSplural</code>	306
<code>\Glsplural</code>	307
<code>\glsplural</code>	306
<code>\glspluralaccessdisplay</code>	148
<code>\glspluralsuffix</code>	129, 185, 189
<code>\glspostdescription</code>	16, 17, 115, 178, 356–366, 368–372
<code>\glspostinline</code>	367
<code>\glspostlinkhook</code> ..	57, 59, 72–75, 88, 190–200
<code>\glsprestandardsort</code>	112
<code>\glsresetentrylist</code>	120, 133
<code>\glssee</code>	43, 45
<code>\glsseeformat</code>	42, 48, 111, 117, 118
<code>\glsseelist</code>	42
<code>\glssetabbrvfmt</code>	56, 64, 85, 86, 163–168, 170, 176, 189–194, 196–198, 200
<code>\glssetattribute</code>	207–211, 213, 215–220, 222–228, 230, 234, 236, 238–241, 243, 245, 248, 250, 252–255, 257–260, 262, 264, 265, 267, 269–271, 274, 275, 277, 278, 280– 282, 284–287, 289, 291–294, 296, 297, 299
<code>\glssetcategoryattribute</code>	95, 107, 109, 140, 146, 158, 159, 161, 162, 177
<code>\glssetnoexpandfield</code>	12, 13
<code>\glssettoctitle</code>	114
<code>\glsshortaccessdisplay</code>	152
<code>\glsshortpltok</code>	186, 187, 207– 211, 213, 215, 217, 223, 225–228, 234, 235, 238, 239, 241, 243, 248, 250, 252– 260, 269, 271, 274, 275, 277, 279–282, 284, 285, 289, 290, 292–294, 296, 297, 299
<code>\glsshortpluralaccessdisplay</code>	153
<code>\glsshorttok</code> 107, 185–187, 206–211, 213, 215, 217, 222–228, 230, 234, 235, 238–241, 243, 245, 248–250, 252, 253, 255–260, 262, 264, 269, 271, 274, 275, 277, 279–282, 284, 285, 287, 289, 290, 292–294, 296–298
<code>\glsesubentryitem</code>	131, 132, 356–366, 368–370, 382, 393
<code>\glsymbolaccessdisplay</code>	150

<code>\glssymbolpluralaccessdisplay</code>	150, 151	<code>\glxtr@label</code>	395
<code>\glstarget</code>	131, 132, 356–366, 368–370, 382, 383, 393, 394	<code>\glxtr@langtag</code>	129
<code>\GLStext</code>	305, 306	<code>\glxtr@linkprefix</code>	129
<code>\Glstext</code>	306	<code>\glxtr@loaddialect</code>	319, 353
<code>\glstext</code>	305	<code>\glxtr@makeglossaries</code>	110
<code>\glstextaccessdisplay</code>	147, 148	<code>\glxtr@newabbreviation</code>	108, 185
<code>\glstextformat</code>	59, 62	<code>\glxtr@next</code>	182
<code>\glstextup</code>	223	<code>\glxtr@org@@do@wrglossary</code>	27
<code>\glstreechildpredesc</code>	368, 369	<code>\glxtr@org@dohyperlink</code>	82
<code>\glstreechildprelocation</code>	368, 369, 371	<code>\glxtr@org@getgrouptitle</code>	119
<code>\glstreegroupheaderfmt</code>	368–371, 383–389, 391	<code>\glxtr@org@newignoredglossary</code>	36
<code>\glstreeindent</code>	369, 370, 381–383	<code>\glxtr@org@makenoidxglossaries</code>	47
<code>\glstreeitem</code>	367, 385, 393	<code>\glxtr@pluralsuffixes</code>	129
<code>\glstreenamebox</code>	382, 383	<code>\glxtr@process</code>	133, 134
<code>\glstreenamefmt</code>	368–370, 372, 374–383	<code>\glxtr@provideignoredglossary</code>	38
<code>\glstreenavigationfmt</code>	369–371, 383–388	<code>\glxtr@punclist</code>	181, 182
<code>\glstreepredesc</code>	368–370	<code>\glxtr@record</code>	11, 129
<code>\glstreeprelocation</code>	367–370, 372	<code>\glxtr@record@nr</code>	13
<code>\glstreesubitem</code>	367, 393	<code>\glxtr@recordsee</code>	12
<code>\glstreesubsubitem</code>	368, 394	<code>\glxtr@resource</code>	127, 129
<code>\GlstrLetField</code>	33	<code>\glxtr@s@newignoredglossary</code>	36
<code>\glstype</code>	59, 61, 72–75, 141, 190–200	<code>\glxtr@s@provideignoredglossary</code>	38
<code>\glunset</code>	47, 59, 98, 99, 141	<code>\glxtr@saveentrycounter</code>	8, 9, 12, 79
<code>\gluserdescription</code>	274, 275, 277, 280	<code>\glxtr@setaccessdisplay</code>	170
<code>\glswrite</code>	43, 110	<code>\glxtr@setbookindexmark</code>	395
<code>\glswriteentry</code>	8, 9	<code>\glxtr@setup@record</code>	13, 14, 26, 27
<code>\Glsxtr</code>	52	<code>\glxtr@shortcutsval</code>	129
<code>\glxtr</code>	52	<code>\glxtr@texencoding</code>	129
<code>\glxtr@@do@wrglossary</code>	8, 10, 12, 13	<code>\glxtr@undefaction@nr</code>	7
<code>\glxtr@addloclistfield</code>	14	<code>\glxtr@undefaction@val</code>	7
<code>\glxtr@addunused</code>	46	<code>\glxtr@usesee</code>	41
<code>\glxtr@applyabrvfmt</code>	200	<code>\glxtr@warnonexistsordo</code>	7, 13, 14, 40
<code>\glxtr@applyabrvstyle</code>	183, 185, 203	<code>\glxtr@writefields</code>	127
<code>\glxtr@counterrecord</code>	130	<code>\glxtr@abbreviationfont</code>	57
<code>\glxtr@do@alsoindex@wrglossary</code>	14	<code>\glxtr@abrvfootnote</code>	211–213, 234–236, 248–250, 269–271
<code>\glxtr@doooption</code>	5, 16, 17, 23, 24, 26	<code>\glxtr@abrvpluralsuffix</code>	129, 189, 207, 209, 212, 214, 215, 217, 220, 223, 237, 251, 274, 283, 286, 289, 294, 297
<code>\glxtr@fields</code>	129	<code>\glxtr@abrvtype</code>	17, 187
<code>\glxtr@headentry@p</code>	89, 90	<code>\glxtr@activatenopost</code>	114
<code>\glxtr@hyperoutsidefalse</code>	60	<code>\glxtr@addallcrossrefs</code>	46
<code>\glxtr@hyperoutsidetrue</code>	60	<code>\glxtr@alias</code>	79
<code>\glxtr@ifnextpunc</code>	182	<code>\glxtr@AltTreeIndent</code>	372
<code>\glxtr@ifpunctoken</code>	182	<code>\glxtr@altrtreeInit</code>	382, 387, 388
<code>\glxtr@inc@linkcount</code>	61, 146	<code>\glxtr@AltTreePar</code>	372
<code>\glxtr@inc@wrglossaryctr</code>	8, 9, 23, 27	<code>\glxtr@AltTreeSetHangIndent</code>	372, 382
<code>\glxtr@indexonly@saveentrycounter</code>	13, 14, 27	<code>\glxtr@AltTreeSetSubHangIndent</code>	383
<code>\glxtr@keylist</code>	50, 51	<code>\glxtr@altrtreeSubSymbolDescLocation</code>	383

<code>\glxtralttreeSymbolDescLocation</code> ..	<code>\glxtrcombingdiacriticIIrules</code> . 325
..... 372, 382	<code>\glxtrcombingdiacriticIIrules</code> .. 325
<code>\glxtrassignfieldfont</code> 64–71	<code>\glxtrcombingdiacriticIrules</code> ... 325
<code>\glxtrautoindex</code> 172	<code>\glxtrcombingdiacriticIVrules</code> .. 325
<code>\glxtrautoindexassignsort</code> 172	<code>\glxtrComputeTreeIndent</code> 382
<code>\glxtrautoindexentry</code> 172	<code>\glxtrComputeTreeSubIndent</code> 382
<code>\glxtrbibaddress</code> 321	<code>\glxtrcounterprefix</code> 122
<code>\glxtrbibauthor</code> 321	<code>\glxtrcurrencyrules</code> 328
<code>\glxtrbibbooktitle</code> 321	<code>\Glsxtrdefaultsubsequentfmt</code> ... 203, 204
<code>\glxtrbibchapter</code> 321	<code>\glxtrdefaultsubsequentfmt</code> ... 202, 204
<code>\glxtrbibedition</code> 321	<code>\Glsxtrdefaultsubsequentplfmt</code> . 203, 204
<code>\glxtrbibhowpublished</code> 321	<code>\glxtrdefaultsubsequentplfmt</code> . 203, 204
<code>\glxtrbibinstitution</code> 321	<code>\GlsXtrDefineAbbreviationShortcuts</code> . 20
<code>\glxtrbibjournal</code> 321	<code>\GlsXtrDefineAcShortcuts</code> 20, 21
<code>\glxtrbibmonth</code> 321	<code>\GlsXtrDefineOtherShortcuts</code> 20
<code>\glxtrbibnote</code> 321	<code>\glxtrdetoklocation</code> 140
<code>\glxtrbibnumber</code> 321	<code>\glxtrdiscardperiod</code> 179
<code>\glxtrbiborganization</code> 321	<code>\glxtrdisplayendloc</code> 121
<code>\glxtrbibpages</code> 321	<code>\glxtrdisplayendloohook</code> 121
<code>\glxtrbibpublisher</code> 321	<code>\glxtrdisplaysingleloc</code> 121
<code>\glxtrbibschool</code> 321	<code>\glxtrdisplaystartloc</code> 121
<code>\glxtrbibseries</code> 321	<code>\glxtrdoautoindexname</code> 80, 81, 169
<code>\glxtrbibtitle</code> 321	<code>\glxtrdopostpunc</code> 213, 236, 250, 271
<code>\glxtrbibtype</code> 321	<code>\glxtrdowrglossaryhook</code> 80
<code>\glxtrbibvolume</code> 321	<code>\GlsXtrDualField</code> 320
<code>\glxtrbookindexatendgroup</code> 393	<code>\glxtremsuffix</code> 252, 254, 256,
<code>\glxtrbookindexatsubendgroup</code> 393	257, 259, 260, 262, 264, 266, 267, 270, 272
<code>\glxtrbookindexatsubsubendgroup</code> .. 393	<code>\GlsXtrEnableEntryCounting</code> 106
<code>\glxtrbookindexbetween</code> 393	<code>\GlsXtrEnableEntryUnitCounting</code> 95
<code>\glxtrbookindexbookmark</code> 394	<code>\GlsXtrEnableOnTheFly</code> 49, 52
<code>\glxtrbookindexcols</code> 392	<code>\glxtrendfor</code> 32
<code>\glxtrbookindexcolspread</code> 392	<code>\glxtrfieldlistgadd</code> 130
<code>\glxtrbookindexfirstmarkfmt</code> 395	<code>\glxtrfieldtitlecase</code> 164–167, 170
<code>\glxtrbookindexformatheader</code> 394	<code>\glxtrfieldtitlecasecs</code> 163
<code>\glxtrbookindexgroupskip</code> 394	<code>\glxtrfieldxifinlist</code> 135
<code>\glxtrbookindexlastmarkfmt</code> 395	<code>\glxtrfirstscfont</code> 223
<code>\glxtrbookindexmulticolseiv</code> 392	<code>\glxtrfirstsmfont</code> 237
<code>\glxtrbookindexname</code> 390, 393	<code>\GlsXtrFmtDefaultOptions</code> 30
<code>\glxtrbookindexparentchildsep</code> 390, 392	<code>\glxtrfmtdisplay</code> 30
<code>\glxtrbookindexparentsubchildsep</code> .	<code>\GlsXtrFmtField</code> 30
..... 392, 393	<code>\glxtrfootnotename</code>
<code>\glxtrbookindexprelocation</code> ... 390, 393 211, 213, 234, 235, 248, 249, 269, 271
<code>\glxtrbookindexsubbetween</code> 393	<code>\GlsXtrFormatLocationList</code> 54, 56, 379–381
<code>\glxtrbookindexsubname</code> 394	<code>\GLSxtrfull</code> 18, 19, 310, 311
<code>\glxtrbookindexsubprelocation</code> 394	<code>\Glsxtrfull</code> 18, 19, 311
<code>\glxtrbookindexsubsubbetween</code> 393	<code>\glxtrfull</code> 18, 19, 310
<code>\glxtrbookindexthepage</code> 395	<code>\Glsxtrfullformat</code>
<code>\glxtrcat</code> 50, 51 188, 202, 204, 205, 207, 209, 212,
<code>\glxtrchecknohyperfirst</code> 65–67	214, 216, 218, 221, 224, 226, 228, 229,

232–234, 236, 238, 240, 242, 243, 246, 247, 249, 250, 252, 254, 256, 258, 260, 261, 263, 265, 267, 269, 270, 272, 275, 276, 278, 281, 284, 287, 290, 292, 295, 297	\glsxtrfullformat	188, 202, 204, 205, 207, 209, 212, 214, 216, 218, 221, 224, 226, 228, 229, 232–234, 236, 238, 240, 242, 243, 246–248, 250, 252, 254, 256, 257, 259, 261, 263, 265, 267, 268, 270, 272, 274, 276, 278, 281, 284, 287, 290, 292, 295, 297
\GLSxtrfullpl	18, 19, 311	
\Glsxtrfullpl	18, 19, 312	
\glsxtrfullpl	18, 19, 311	
\Glsxtrfullplformat	188, 202, 204, 205, 207, 210, 212, 214, 216, 218, 221, 224, 226, 228, 229, 232, 233, 235, 236, 238, 240, 242, 244, 246, 248, 249, 251, 253, 254, 256, 258, 260, 261, 263, 265, 267, 269, 270, 272, 275, 276, 278, 281, 284, 287, 290, 292, 295, 298	
\glsxtrfullplformat	201, 202, 204, 205, 207, 209, 212, 214, 216, 218, 221, 224, 226, 228, 229, 232–234, 236, 238, 240, 242, 243, 246–248, 250, 252, 254, 256, 258, 260, 261, 263, 265, 267, 269, 270, 272, 275, 276, 278, 281, 284, 287, 290, 292, 295, 297	
\glsxtrfullsep	187, 188, 207–210, 212–219, 221, 223–231, 233, 235, 237–247, 249, 251–268, 270–273, 283–288, 291–294, 298	
\glsxtrgenabbrvfmt	57	
\glsxtrgeneralpuncIIrules	328	
\glsxtrgeneralpuncIrules	328	
\glsxtrgetgrouptitle	120, 394	
\glsxtrgroupfield	136	
\Glsxtrheadfirst	302	
\glsxtrheadfirst	302	
\Glsxtrheadfirstplural	302	
\glsxtrheadfirstplural	302	
\Glsxtrheadfull	302	
\glsxtrheadfull	302	
\Glsxtrheadfullpl	302	
\glsxtrheadfullpl	302	
\Glsxtrheadlong	302	
\glsxtrheadlong	302	
\Glsxtrheadlongpl	302	
\glsxtrheadlongpl	302	
\Glsxtrheadname	301	
\glsxtrheadname	131, 301	
\Glsxtrheadplural	302	
\glsxtrheadplural	302	
\Glsxtrheadshort	301	
\glsxtrheadshort	301	
\Glsxtrheadshortpl	301	
\glsxtrheadshortpl	301	
\Glsxtrheadtext	302	
\glsxtrheadtext	301	
\glsxtrhyperlink	23, 24, 83	
\glsxtrhyphensuffix	284, 292	
\glsxtrifcounttrigger	98, 99	
\glsxtrifcustomdiscardperiod	179	
\glsxtrifemptyglossary	120, 126, 133	
\glsxtrifhasfield	34, 79, 320, 390	
\glsxtrifhyphenstart	283, 285, 288, 291, 293, 294	
\glsxtrifindexing	80	
\glsxtrifinmark	63, 89–92, 300–302	
\glsxtrifnextpunc	182	
\glsxtrifperiod	179, 181	
\glsxtrifrecordtrigger	142–144	
\glsxtrifwasfirstuse	64–67, 71–75, 78, 109, 180, 190, 193–200, 213, 236, 250, 271, 272, 275, 277–279, 289, 291, 294, 296	
\glsxtrinclinkcounter	146	
\glsxtrinindexaliased	79	
\glsxtrinindexseealso	44, 45	
\glsxtrinithyperoutside	61	
\glsxtrinitwrgloss	61, 141	
\glsxtrinitwrglossbeforefalse	59, 60	
\glsxtrinitwrglossbeforetrue	59	
\Glsxtrinlinefullformat	188, 190, 204, 205, 212, 214, 215, 217, 219, 221, 227, 229–231, 233, 235, 237, 242–245, 247, 249, 251, 259, 261–263, 265, 266, 268, 271, 272, 276, 279, 287, 290, 295, 298, 317	
\glsxtrinlinefullformat	188, 189, 191, 204, 205, 212, 214, 215, 217, 219, 221, 227, 229– 231, 233, 235, 237, 241, 243–245, 247, 249, 251, 259–261, 263, 264, 266, 268, 270, 272, 276, 278, 286, 290, 295, 298, 317	
\Glsxtrinlinefullplformat	188, 192, 204, 205, 213, 214, 216, 218, 219, 221, 228– 231, 233, 235, 237, 242–244, 246, 247, 249, 251, 259, 261–263, 265, 267, 268, 271, 272, 276, 279, 287, 290, 295, 298, 318	

<code>\glxstrinlinefullplformat</code>	<code>\glxstrmarkhook</code>
..... 188, 191, 192, 204, 205,	299, 300
212, 214, 215, 217, 219, 221, 227, 229–	<code>\glxstrMathItalicAlpha</code>
231, 233, 235, 237, 242–245, 247, 249,	339, 343, 344
251, 259, 260, 262, 263, 265, 266, 268,	<code>\glxstrMathItalicBeta</code>
270, 272, 276, 279, 286, 290, 295, 298, 318	339, 343, 344
<code>\glxstrinsertinsidefalse</code>	<code>\glxstrMathItalicChi</code> ...
206	339, 340, 344, 345
<code>\GlsXtrInternalLocationHyperlink</code> 23, 122	<code>\glxstrMathItalicDelta</code>
<code>\glxstrLatinA</code>	339, 343, 344
330–334	<code>\glxstrMathItalicEpsilon</code> 339, 340, 343, 344
<code>\glxstrLatinAELigature</code>	<code>\glxstrMathItalicEta</code> ...
331, 334	339, 340, 343, 344
<code>\glxstrLatinE</code>	<code>\glxstrMathItalicGamma</code>
330–334	339, 343, 344
<code>\glxstrLatinEszettSs</code>	<code>\glxstrMathItalicIota</code> ..
331–333, 335	339, 340, 343, 344
<code>\glxstrLatinEszettSz</code>	<code>\glxstrMathItalicKappa</code> .
331, 333	339, 340, 343, 344
<code>\glxstrLatinEth</code>	<code>\glxstrMathItalicLambda</code> 339, 340, 343, 344
330–334	<code>\glxstrMathItalicMu</code>
<code>\glxstrLatinH</code>	339, 340, 343, 344
330–334	<code>\glxstrMathItalicNu</code>
<code>\glxstrLatinI</code>	339, 340, 343, 344
330–334	<code>\glxstrMathItalicOmega</code> .
<code>\glxstrLatinInsularG</code>	339, 340, 344, 345
334	<code>\glxstrMathItalicOmicron</code> 339, 340, 343, 344
<code>\glxstrLatinK</code>	<code>\glxstrMathItalicPhi</code> ...
330–335	339, 340, 343, 345
<code>\glxstrLatinL</code>	<code>\glxstrMathItalicPi</code>
330–335	339, 340, 343, 344
<code>\glxstrLatinM</code>	<code>\glxstrMathItalicPsi</code> ...
330–335	339, 340, 344, 345
<code>\glxstrLatinN</code>	<code>\glxstrMathItalicRho</code> ...
330–335	339, 340, 343, 344
<code>\glxstrLatinO</code>	<code>\glxstrMathItalicSigma</code> .
330–335	339, 340, 343, 344
<code>\glxstrLatinOELigature</code>	<code>\glxstrMathItalicTau</code> ...
332, 334, 335	339, 340, 343, 344
<code>\glxstrLatinP</code>	<code>\glxstrMathItalicTheta</code> .
330–335	339, 340, 343, 345
<code>\glxstrLatinS</code>	<code>\glxstrMathItalicUpsilon</code> 339, 340, 343, 345
330–335	<code>\glxstrMathItalicXi</code>
<code>\glxstrLatinT</code>	339, 340, 343, 344
330–335	<code>\glxstrMathItalicZeta</code> ..
<code>\glxstrLatinThorn</code>	339, 340, 343, 344
334	<code>\glxstrnewabbrevpresetkeyhook</code>
<code>\glxstrLatinX</code>	186
330–335	<code>\glxstrnewnumber</code>
<code>\glxstrlocationhyperlink</code>	19
122	<code>\glxstrnewsymbol</code>
<code>\glxstrlocrangefmt</code>	19
121	<code>\glxstrNoGlossaryWarning</code>
<code>\GLSxtrlong</code>	21, 123
18, 19, 308, 309	<code>\GlsXtrNoGlsWarningAutoMake</code>
<code>\Glsxtrlong</code>	127
18, 19, 309, 310	<code>\GlsXtrNoGlsWarningBuildInfo</code>
<code>\glxstrlong</code>	127
18, 19, 309	<code>\GlsXtrNoGlsWarningCheckFile</code>
<code>\glxstrlonghyphen</code>	126
290	<code>\GlsXtrNoGlsWarningEmptyMain</code> ..
<code>\glxstrlonghyphennoshort</code>	126, 127
286, 287	<code>\GlsXtrNoGlsWarningEmptyNotMain</code> ...
<code>\glxstrlonghyphenshort</code>	126
284	<code>\GlsXtrNoGlsWarningEmptyStart</code>
<code>\glxstrlongnoshortdescname</code> .	126
220, 267, 286	<code>\GlsXtrNoGlsWarningHead</code>
<code>\glxstrlongnoshortname</code>	126
.....	<code>\GlsXtrNoGlsWarningMisMatch</code>
222, 230, 245, 262, 264, 287	127
<code>\GLSxtrlongpl</code>	<code>\GlsXtrNoGlsWarningNoOut</code>
18, 19, 309, 310	127
<code>\Glsxtrlongpl</code>	<code>\GlsXtrNoGlsWarningTail</code>
18, 19, 310	127
<code>\glxstrlongpl</code>	<code>\glxstrnopostpunc</code>
18, 19, 309	114
<code>\glxstrlongshortdescname</code>	<code>\glxstronlydescname</code>
.....	298
208, 224, 239, 253, 255, 284, 290	<code>\glxstronlydescsort</code>
<code>\glxstrlongshortdescsort</code>	298
.....	<code>\glxstronlyname</code>
208, 224, 239, 253, 255, 280, 285, 290	297
<code>\glxstrlongshortname</code>	<code>\glxstronlysuffix</code>
.....	297
206, 223, 238, 252, 253, 274, 275, 284, 289	<code>\glxstrorg@ifKV@glslink@hyper</code>
<code>\glxstrlongshortuserdescname</code> ..	57
277, 280	<code>\glxstrorglong</code>
	185, 208, 286
	<code>\glxstrorgshort</code>
	185, 208
	<code>\GLSxtrp</code>
	89
	<code>\Glsxtrp</code>
	89

<code>\glxtrp</code>	88, 90	<code>\glxtrsetupfulldefs</code>	
<code>\glxtrparen</code>	180,	190–192, 213, 236, 250, 272
187, 188, 207–210, 212–219, 221, 223–		<code>\GLSxtrshort</code>	18, 19, 92, 303, 304
231, 233, 235, 237–247, 249, 251–263,		<code>\Glsxtrshort</code>	18, 19, 304
265–268, 270–273, 283–288, 291–294, 298		<code>\glxtrshort</code>	18, 303
<code>\Glsxtrpl</code>	52	<code>\glxtrshortdescname</code> ...	217, 228, 242, 260
<code>\glxtrpl</code>	52	<code>\glxtrshorthyphen</code>	295
<code>\glxtrpostdescription</code> .	115, 161, 178, 367	<code>\glxtrshorthyphenlong</code>	292
<code>\glxtrposthyphenlong</code>	294, 296	<code>\glxtrshortlongdescname</code>	
<code>\glxtrposthyphenshort</code>	289, 291	210, 226, 240, 256, 258, 293, 296
<code>\glxtrposthyphensubsequent</code>		<code>\glxtrshortlongdescsort</code>	
.....	289, 291, 294, 296	210, 226, 240, 256, 258, 282, 293, 296
<code>\glxtrpostlink</code>	179	<code>\glxtrshortlongname</code>	
<code>\glxtrpostlinkendsentence</code>	179	209, 225, 239, 255, 257, 277, 280, 292, 294
<code>\glxtrpostlinkhook</code>	179	<code>\glxtrshortlonguserdescname</code> ..	279, 282
<code>\glxtrpostlocalreset</code>	94, 96, 104	<code>\glxtrshortnolongname</code> .	215, 227, 241, 259
<code>\glxtrpostlocalunset</code>	94, 96, 104	<code>\GLSxtrshortpl</code>	18, 19, 303, 304
<code>\glxtrpostlongdescription</code>	36	<code>\Glsxtrshortpl</code>	18, 19, 304
<code>\glxtrpostnamehook</code>	166–168, 171	<code>\glxtrshortpl</code>	18, 19, 303
<code>\GlsXtrPostNewAbbreviation</code> .	187, 204,	<code>\glxtrsmfont</code>	237
205, 207–211, 213, 215–220, 222, 223,		<code>\glxtrsmsuffix</code>	
225–228, 230, 234, 235, 238–241, 243,		238, 240, 241, 243, 245, 246, 248, 250
245, 248, 250, 252–255, 257–260, 262,		<code>\Glsxtrsubsequentfmt</code>	
264, 265, 267, 269, 271, 274, 275, 277–		201, 204, 220, 231, 232,
282, 284–287, 289, 291–294, 296, 297, 299		245, 247, 263, 264, 266, 268, 286, 289, 295
<code>\glxtrpostreset</code>	94, 96, 104	<code>\glxtrsubsequentfmt</code>	
<code>\glxtrpostunset</code>	93, 96, 104	201, 204, 220, 231, 232,
<code>\glxtrprelocation</code>	245, 246, 262, 264, 266, 268, 286, 289, 295
.....	357, 359, 361, 363, 365, 367, 390	<code>\Glsxtrsubsequentplfmt</code>	
<code>\glxtrprotectlinks</code>	82–84	201, 204, 220, 231, 232,
<code>\GlsXtrRecordCounter</code>	11	245, 247, 263, 264, 266, 268, 286, 289, 295
<code>\glxtrrecordtriggervalue</code>	141	<code>\glxtrsubsequentplfmt</code>	
<code>\glxtrregularfont</code>	56, 57, 64	201, 204, 220, 231, 232,
<code>\glxtrresourcecount</code>	128	245, 247, 262, 264, 266, 268, 286, 289, 295
<code>\glxtrresourcefile</code>	128	<code>\glxtrsupplocationurl</code>	122, 123
<code>\glxtrresourceinit</code>	127	<code>\glxtrtagfont</code>	178
<code>\glxtrrestoremarkhook</code>	300	<code>\Glsxtrtitlefirst</code>	301, 302, 315
<code>\glxtrrestorepostpunc</code>	115	<code>\glxtrtitlefirst</code>	301, 302, 315
<code>\glxtrscfont</code>	223	<code>\Glsxtrtitlefirstplural</code>	301, 302, 316
<code>\glxtrscsuffix</code>		<code>\glxtrtitlefirstplural</code> ..	301, 302, 315, 316
....	224, 225, 227, 228, 231, 232, 234, 236	<code>\Glsxtrtitlefull</code>	301, 302, 317, 318
<code>\GlsXtrSetActualChar</code>	175	<code>\glxtrtitlefull</code>	301, 302, 317
<code>\glxtrsetaliasnoindex</code>	13, 14, 79	<code>\Glsxtrtitlefullpl</code>	301, 302, 318
<code>\GlsXtrSetEncapChar</code>	175	<code>\glxtrtitlefullpl</code>	301, 302, 318
<code>\GlsXtrSetEscChar</code>	175	<code>\Glsxtrtitlelong</code>	301, 302, 316
<code>\glxtrsetfieldifexists</code>	33, 34	<code>\glxtrtitlelong</code>	301, 302, 316
<code>\GlsXtrSetLevelChar</code>	175	<code>\Glsxtrtitlelongpl</code>	301, 302, 317
<code>\glxtrsetpopts</code>	88	<code>\glxtrtitlelongpl</code>	301, 302, 317
		<code>\Glsxtrtitlename</code>	301, 302, 313, 314

<code>\glxtrtrtitlename</code>	301, 302, 313	<code>\glxtruserfield</code>	273
<code>\glxtrtrtitleorpdforheading</code>		<code>\glxtruserparen</code>	274–282
.....	25, 131, 132, 301, 302	<code>\glxtrusersuffix</code>	274, 276, 278, 281
<code>\Glsxtrtrtitleplural</code>	301, 302, 315	<code>\glxtruseseealsoformat</code>	42, 43
<code>\glxtrtrtitleplural</code>	301, 302, 314	<code>\glxtruseseeeformat</code>	41
<code>\Glsxtrtrtitleshort</code>	301, 302, 313	<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	183, 205
<code>\glxtrtrtitleshort</code>	301, 302, 312	<code>\GlsXtrWarning</code>	50, 51
<code>\Glsxtrtrtitleshortpl</code>	301, 302, 313	<code>\glxtrword</code>	184
<code>\glxtrtrtitleshortpl</code>	301, 302, 312	<code>\glxtrwordsep</code>	184, 283, 285, 288, 291, 293, 294
<code>\Glsxtrtrtitledtext</code>	301, 302, 314	<code>\glxtrwrglossmark</code>	24
<code>\glxtrtrtitledtext</code>	301, 302, 314		
<code>\GlsXtrTotalRecordCount</code>	140		
<code>\glxtrtreetopindent</code>	372, 381		
<code>\glxtrundefaction</code>	7, 13, 14, 28, 36, 37, 39, 40		
<code>\glxtrundeftag</code>	27, 117, 118		
<code>\glxtrunsrtdo</code>	134, 135		
<code>\glxtrUpAlpha</code>	338, 343, 344		
<code>\glxtrUpBeta</code>	338, 343, 344		
<code>\glxtrUpChi</code>	338, 339, 344, 345		
<code>\glxtrUpDelta</code>	338, 343, 344		
<code>\glxtrUpDigamma</code>	338, 339, 343		
<code>\glxtrUpEpsilon</code>	338, 343, 344		
<code>\glxtrUpEta</code>	338, 343, 344		
<code>\glxtrUpGamma</code>	338, 343, 344		
<code>\glxtrUpIota</code>	338, 343, 344		
<code>\glxtrUpKappa</code>	338, 343, 344		
<code>\glxtrUpLambda</code>	338, 343, 344		
<code>\glxtrUpMu</code>	338, 343, 344		
<code>\glxtrUpNu</code>	338, 343, 344		
<code>\glxtrUpOmega</code>	338, 339, 344, 345		
<code>\glxtrUpOmicron</code>	338, 339, 343, 344		
<code>\glxtrUpPhi</code>	338, 339, 344, 345		
<code>\glxtrUpPi</code>	338, 339, 343, 344		
<code>\glxtrUpPsi</code>	338, 339, 344, 345		
<code>\glxtrUpRho</code>	338, 339, 343, 344		
<code>\glxtrUpSigma</code>	338, 339, 343, 344		
<code>\glxtrUpTau</code>	338, 339, 343, 344		
<code>\glxtrUpTheta</code>	338, 343, 344		
<code>\glxtrUpUpsilon</code>	338, 339, 343, 345		
<code>\glxtrUpXi</code>	338, 339, 343, 344		
<code>\glxtrUpZeta</code>	338, 343, 344		
<code>\GlsXtrUseAbbrStyleFmts</code>			
.....	208, 210, 217–219,		
	222, 223, 225, 227, 230, 239, 241, 244,		
	253, 255, 257, 258, 261, 266, 269, 277,		
	280, 282, 285, 286, 288, 291, 293, 296, 299		
<code>\GlsXtrUseAbbrStyleSetup</code>			
.....	216, 218, 219, 221,		
	222, 230, 232, 244, 246, 261, 265, 266, 269		

H

<code>\hangindent</code> .	369, 370, 372, 381–383, 387, 388
<code>\hbox</code>	356
<code>\hfill</code>	356
<code>\href</code>	82
<code>\hsize</code>	53
<code>\hss</code>	356
<code>\hyperlink</code>	83
<code>\hyperpage</code>	171
<code>\hyperref</code>	82, 122, 320
<code>hyperref package</code>	84, 171, 299, 312

I

<code>\if</code>	49
<code>\if@glxtr@autoseeindex</code> . . .	25, 26, 41, 44
<code>\if@glxtr@format@override</code>	172
<code>\if@glxtrdocdefrestricted</code>	48
<code>\if@glxtrindexcrossrefs</code>	15, 46
<code>\ifblank</code>	28, 50, 51, 110
<code>\ifcase</code> . .	7, 13, 20, 21, 24, 48, 59, 116, 368, 393
<code>\ifcsdef</code>	27,
	34–39, 60, 62, 76, 77, 88–92, 101, 113,
	119, 120, 131, 135, 138–140, 145, 163–
	166, 168–170, 180, 183, 200, 204, 359–366
<code>\ifcsstring</code>	28, 159, 203
<code>\ifcsundef</code>	32, 34–39, 48, 52,
	55, 84, 96, 101–105, 118, 119, 123, 135,
	146, 159, 203–206, 356, 373, 374, 381, 395
<code>\ifcsvoid</code>	45, 158
<code>\ifdef</code>	14,
	19, 26, 30, 40, 42, 43, 47, 53, 78, 82, 83,
	89–91, 113, 117, 118, 129, 138, 161, 162,
	175, 178, 273, 301, 312–318, 320, 353,
	356–359, 367–371, 383–388, 391, 394, 395
<code>\ifdefempty</code>	7–9, 32,
	36–38, 41, 42, 61, 63, 95, 107, 110, 113,
	121, 133, 136, 140, 141, 177, 184, 200, 392
<code>\ifdefequal</code>	47, 127, 136, 169
<code>\ifdefstring</code>	6, 23, 34, 172, 177, 391

<code>\ifdefvoid</code>	41, 44–46, 83, 101, 118, 122, 136, 137
<code>\ifdim</code>	53, 54, 109, 373–381
<code>\IfFileExists</code>	22, 123, 126, 127, 130, 353, 355
<code>\ifglossaryexists</code>	40
<code>\ifglshasacronym</code>	17, 126
<code>\ifglshasacrshortcuts</code>	20
<code>\ifglshasautomake</code>	113, 126, 130
<code>\ifglshasentrycounter</code>	34
<code>\ifglshasentryexists</code>	9, 39, 40, 50, 51, 54, 64, 136, 159, 178, 179
<code>\ifglshasfield</code>	157
<code>\ifglshaslong</code>	30, 273
<code>\ifglshasparent</code>	100, 101, 144, 145
<code>\ifglshasparent</code>	131–133, 136, 374, 376, 377
<code>\ifglshasshort</code>	42, 56, 57, 64
<code>\ifglshassymbol</code>	180, 368–370, 372
<code>\ifglshasindexonlyfirst</code>	80
<code>\ifglshasgroupskip</code>	357, 359–369, 371, 383, 391
<code>\ifglshasnonumberlist</code>	56
<code>\ifglshasnopostdot</code>	16, 115
<code>\ifglshassanitizesort</code>	112
<code>\ifglshassubentrycounter</code>	34
<code>\ifglshasused</code>	46, 78, 80, 97, 106, 109, 200, 374–376, 378–380
<code>\ifglshasxindy</code>	123, 125
<code>\ifglshasxtr@hyperoutside</code>	62
<code>\ifglshasxtrinitwrglossbefore</code>	59, 62, 141
<code>\ifglshasxtrininsertinside</code>	193–200, 202, 203, 207, 209, 210, 212–221, 224, 226– 254, 256–272, 274–276, 278, 279, 281, 283, 285, 288, 290, 291, 294, 295, 297, 298
<code>\ifHy@hyperindex</code>	171
<code>\ifinlistcs</code>	31, 48
<code>\ifinner</code>	25
<code>\ifKV@glshlink@hyper</code>	57, 61–63
<code>\ifKV@glshlink@local</code>	59, 141
<code>\ifKV@glshlink@noindex</code>	8, 9, 12, 30, 79, 80
<code>\ifmmode</code>	25
<code>\ifnum</code>	15, 47, 97, 105, 106, 118, 128, 141, 369, 370, 382
<code>\ifstrempty</code>	131, 138, 146
<code>\ifstrequal</code>	16, 17, 21, 22
<code>\ifthenelse</code>	126, 127
<code>\IfTrackedLanguageFileExists</code>	319
<code>\ifundef</code>	23, 32, 110, 177, 178, 353
<code>\ifx</code>	8–11, 43, 53, 54, 62, 110, 114, 121, 122, 130, 172, 173, 175, 182, 184, 186, 282, 283, 389
<code>\immediate</code>	97, 105, 123, 130
<code>\index</code>	172
<code>\indexspace</code>	357, 368–371, 383–389, 391, 394
<code>\input</code>	318
<code>\inputencodingname</code>	129
<code>\InputIfFileExists</code>	47
<code>\istfilename</code>	110
<code>\item</code>	125, 126, 356–359, 367–369, 384, 385
J	
<code>\jobname</code>	47, 123, 125–128, 130
K	
<code>\key@ifundefined</code>	12, 13, 28, 29, 76, 133, 136
<code>\KV@glshlink@hyperfalse</code>	65, 78, 83, 84
<code>\KV@glshlink@hypertrue</code>	84
<code>\KV@glshlink@noindexfalse</code>	78, 79
<code>\KV@glshlink@noindextrue</code>	79, 84
L	
<code>\L</code>	335, 338
<code>\l</code>	338
<code>\label</code>	23
<code>\LaTeX</code>	125, 126
<code>\leaders</code>	356
<code>\leavevmode</code>	36, 61
<code>\let</code>	5, 7–14, 16, 18, 19, 25–27, 30, 32, 35, 36, 47, 49, 52–55, 57–59, 61–75, 77– 79, 81–85, 88, 93–96, 104–111, 113–120, 127–130, 133, 134, 136, 141, 146, 164, 165, 167–173, 177, 178, 182–186, 189– 200, 202–204, 213, 236, 250, 272, 299– 302, 353, 367, 368, 372, 374, 385, 392–395
<code>\letabbreviationstyle</code>	213, 215, 216, 218, 221, 222, 228, 229, 242, 244, 260, 261
<code>\letcs</code>	28, 32, 41, 42, 46, 60, 62, 76, 117–119, 135, 136, 163–171, 395
<code>\levelchar</code>	175
<code>\listadd</code>	101
<code>\listbreak</code>	177
<code>\listcsadd</code>	31
<code>\listcseadd</code>	31, 102
<code>\listcsgadd</code>	31, 48
<code>\listcsxadd</code>	31, 102
<code>\listxadd</code>	93
<code>\loadglshentries</code>	48, 124
<code>\long</code>	35, 36
M	
<code>\MakeAcronymsAbbreviations</code>	109
<code>\makeatletter</code>	47, 123, 128, 174

<code>\makeatother</code>	174	274, 276, 279, 283, 285, 288, 291, 293,
<code>\makebox</code>	356, 382, 383	296–298, 300–321, 324–352, 355, 357,
<code>\makefirstuc</code>	177	367, 371–374, 381, 382, 390, 391, 394, 395
<code>makeglossaries</code>	116	<code>\newcount</code>
<code>\makeglossaries</code>	110, 124–127, 130	128, 137
<code>\makeglossary</code>	110	<code>\newcounter</code>
<code>makeindex</code>	396	23, 145
<code>makeindex</code>	14, 109	<code>\newentry</code>
<code>\makenoidxglossaries</code>	125	19
<code>\MakeTextUppercase</code>	301	<code>\newglossary</code>
<code>\MakeUppercase</code>	301, 302	17, 110
<code>\marginpar</code>	25	<code>\newglossaryentry</code>
<code>\markboth</code>	300 19, 48, 49, 95, 103, 107, 161, 162, 187
<code>\markright</code>	300	<code>\newglossaryentry options</code>
<code>\mathit</code>	322	alias
<code>\mathrm</code>	321–323	16, 41, 43–46
<code>\maxdimen</code>	53, 54	desc
<code>\mbox</code>	358, 359, 382	151, 156
<code>\medskip</code>	126, 127, 135	descplural
<code>\MessageBreak</code> 48, 52, 97, 106, 110, 113, 114, 203		151, 152, 156
<code>mfistuc package</code>	177	first 82, 148, 149, 155, 206, 307, 308, 315, 396
<code>\mfistucMakeUppercase</code> . 64–75, 77, 86,		firstplural
87, 89, 92, 100, 108, 145, 147–157, 166,		149, 155, 206, 308, 315, 396
167, 170, 191, 192, 194, 196, 198, 200–202		group
<code>\mfu@checkword@arg</code>	177	136
<code>\mfu@checkword@do</code>	177	loclist
		31
		long
		153, 157, 316
		longplural
		154, 157, 316
		name
		42, 147, 154, 172, 304, 305, 313
		plural
		148, 155, 206, 306, 307, 314
		see
		15, 16, 26, 41, 43, 46, 49, 111
		seealso
		16, 41–43, 45, 46, 407
		short
		152, 156, 183
		shortplural
		153, 157, 183
		symbol
		150, 155
		symbolplural
		150, 151, 156
		text
		82,
		147, 148, 154, 155, 206, 208, 305, 306, 314
		<code>\newglossarystyle</code>
		392
		<code>\newif</code>
		59, 171, 206
		<code>\newlength</code>
		372
		<code>\newnum</code>
		19
		<code>\newrobustcmd</code> 29, 30, 32–34, 42, 43, 60, 76,
		77, 88, 89, 99, 100, 115, 119, 131, 132,
		134, 135, 138, 139, 142–144, 170, 177,
		178, 189–200, 282, 300, 303–312, 374–380
		<code>\newsym</code>
		19
		<code>\newterm</code>
		161
		<code>\newtoks</code>
		183
		<code>\newwrite</code>
		110
		<code>\nobreak</code>
		358, 359, 368–371, 384–387
		<code>\NoCaseChange</code>
		89–92, 132, 303–311
		<code>\noexpand</code> ... 10, 11, 22, 42, 44, 45, 47, 107,
		123, 127, 134–136, 145, 173, 187, 355, 393
		<code>\nofiles</code>
		126
		<code>\noindent</code>
		126, 127, 370, 371, 385–388
		<code>\nopagebreak</code>
		368–371, 383–390, 394
		<code>\nopostdesc</code>
		36, 50, 51, 114, 161

N

<code>\NeedsTeXFormat</code>	5, 319, 355, 390
<code>\new@glossaryentry</code>	49, 113
<code>\new@ifnextchar</code>	29,
76, 77, 99, 100, 138, 142–144, 181, 189–199	
<code>\newabbr</code>	18, 19
<code>\newabbreviation</code>	18, 19
<code>\newabbreviationhook</code>	186
<code>\newabbreviationstyle</code>	206, 208–
211, 213, 215–228, 230, 232, 234, 235,	
238–242, 244, 246, 248, 249, 252–262,	
264–267, 269, 271, 274, 275, 277, 279,	
280, 282–285, 287, 289–292, 294, 296–298	
<code>\newacronym</code>	107, 108
<code>\newacronymhook</code>	107
<code>\newacronymstyle</code>	108, 109
<code>\newcommand</code>	5–42, 44, 46, 47,
49–52, 54–57, 59, 60, 63–65, 76, 77, 79–	
81, 83, 84, 87–97, 99–109, 113–119, 121,	
122, 124–136, 138–163, 169, 171–185,	
187–200, 202–206, 208, 210, 211, 215,	
217, 220, 222, 223, 237, 251, 252, 273,	

<code>\ns@GLSxtrfull</code>	190	<code>indexcounter</code>	320
<code>\ns@Glsxtrfull</code>	190	<code>nonumberlist</code>	54
<code>\ns@glxtrfull</code>	189	<code>nopostdot</code>	16
<code>\ns@GLSxtrfullpl</code>	192	false	16
<code>\ns@Glsxtrfullpl</code>	191	<code>numbers</code>	19
<code>\ns@glxtrfullpl</code>	191	<code>postdot</code>	16
<code>\ns@GLSxtrlong</code>	196	<code>record</code>	7, 13, 47, 57, 110, 127, 189, 191–200, 405
<code>\ns@Glsxtrlong</code>	195	alsoindex	8, 10
<code>\ns@glxtrlong</code>	194	only	8
<code>\ns@GLSxtrlongpl</code>	199	<code>shortcuts</code>	20
<code>\ns@Glsxtrlongpl</code>	199	ac	20
<code>\ns@glxtrlongpl</code>	198	all	20
<code>\ns@GLSxtrshort</code>	194	false	20
<code>\ns@Glsxtrshort</code>	193	none	20
<code>\ns@glxtrshort</code>	192, 193	true	20
<code>\ns@GLSxtrshortpl</code>	197	<code>sort</code>	
<code>\ns@Glsxtrshortpl</code>	197	use	63
<code>\ns@glxtrshortpl</code>	196	<code>style</code>	22
<code>\null</code>	21	<code>stylemods</code>	22
<code>\number</code>	47, 102–105, 127, 138	<code>symbols</code>	19, 162
<code>\numexpr</code>	102, 105	<code>undefaction</code>	39, 40
O			
<code>\O</code>	335, 337	error	6
<code>\o</code>	337	warn	6
<code>\or</code>	7, 13, 14, 20, 21, 24, 49, 60, 368, 393	<code>xindy</code>	42, 43
<code>\org@glossaryentrynumbers</code>	54, 114	<code>\PackageError</code>	6, 11, 22, 26, 48, 52, 60, 76, 77, 79, 88, 89, 95, 96, 103, 105, 106, 108, 110, 112, 113, 120, 130, 134, 135, 138, 203–206, 355, 356
<code>\org@glossarytitle</code>	114	<code>\PackageWarning</code>	16
<code>\org@ifKV@glslink@hyper</code>	61, 63	<code>\PackageWarningNoLine</code>	16
P			
<code>\p@gl@hyp@opt</code>	81	<code>\pageref</code>	23, 34
<code>package options:</code>		<code>\par</code>	126, 127, 358, 359, 368–372, 382–388, 391
<code>abbreviations</code>	17, 18	<code>\parindent</code>	367, 369, 370, 372, 382, 383, 385–389, 392
<code>accsupp</code>	21, 147	<code>\parskip</code>	367, 369, 370, 385–387, 392
<code>acronym</code>	17	<code>\PassOptionsToPackage</code>	5, 22
<code>automake</code>	113, 125, 130	<code>\pdfbookmark</code>	391
<code>true</code>	130	<code>\preglossarypreamble</code>	34
<code>autoseeindex</code>	26	<code>\preto</code>	79
<code>false</code>	25	<code>\print@noop@unsrtglossaryunit</code>	12, 13
<code>counter</code>		<code>\print@op@unsrtglossaryunit</code>	14
<code>wrglossary</code>	23	<code>\printabbreviations</code>	17
<code>debug</code>		<code>\printglossaries</code>	111, 125
<code>showtargets</code>	83	<code>\printglossary</code>	17, 111, 125
<code>docdef</code>	14, 47, 48, 95, 103	<code>\printglossary options</code>	
<code>false</code>	47, 48	<code>nonumberlist</code>	56
<code>restricted</code>	15	<code>type</code>	113
<code>true</code>	47, 49	<code>\printnoidxglossaries</code>	125
<code>docdefs</code>		<code>\printnoidxglossary</code>	111, 112, 125
<code>restricted</code>	48		

