

glossaries-extra.sty v1.13: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2017-02-07

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)	4
1.1 Package Initialisation and Options	4
1.2 Extra Utilities	16
1.3 Modifications to Commands Provided by glossaries	22
1.3.1 Existence Checks	27
1.3.2 Document Definitions	30
1.3.3 Existing Glossary Style Modifications	35
1.3.4 Entry Formatting, Hyperlinks and Indexing	39
1.3.5 Entry Counting	70
1.3.6 Acronym Modifications	83
1.3.7 Indexing and Displaying Glossaries	85
1.4 Integration with glossaries-accsupp	104
1.5 Categories	115
1.6 Abbreviations	138
1.6.1 Abbreviation Styles Setup	155
1.6.2 Predefined Styles (Default Font)	158
1.6.3 Predefined Styles (Small Capitals)	170
1.6.4 Predefined Styles (Fake Small Capitals)	174
1.6.5 Predefined Styles (Emphasized)	178
1.6.6 Predefined Styles (User Parentheses Hook)	184
1.7 Using Entries in Headings	192
1.8 Multi-Lingual Support	209
2 Style Adjustments (glossaries-extra-stylemods.sty)	211
2.1 Package Initialisation	211
2.2 List-Like Styles	212
2.3 Longtable Styles	212
2.4 Long Ragged Styles	213
2.5 Supertabular Styles	215
2.6 Super Ragged Styles	216
2.7 Inline Style	217
2.8 Tree Styles	217
Glossary	229
Change History	230
Index	240

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2017/02/07 v1.13 (NLCT)]

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

Requires etoolbox package.
4 \RequirePackage{etoolbox}

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

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

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

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

Declare package options.

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

`\f@for@gl@sentries`

```

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

```

`\f@for@gl@sentries`

```

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

```

```

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

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

```

In the event that someone wants to develop a post-processor that needs to know what entries have been used in the document, v1.08 introduces the record option, which will write information to the aux file whenever an entry needs to be indexed.

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

```
76 \newcommand*{\@glstr@record}[2]{}
```

`@@glstr@record` This is the actual code that does the recording 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.

```

77 \newcommand*{\@@glstr@record}[2]{%
78   \begingroup
79   \def\@glsnumberformat{glsnumberformat}%
80   \ifcsdef{glo@#2@counter}%
81     {%
82       \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
83     }%
84     {%

```

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

```

85     \def\@gls@counter{page}%
86   }%
87   \setkeys{glslink}{#1}%
88   \ifKV@glslink@noindex
89     \else
90     \glswriteentry{#2}%
91     {%

```

Save the entry counter.

```
92     \glstr@saveentrycounter
```

Temporarily redefine \@@do@@wrglossary so we can use \glxstr@@do@@wrglossary.

```
93 \let\@@do@@wrglossary\glxstr@dorecord
94 \glxstr@@do@@wrglossary{#2}%
95 }%
96 \fi
97 \endgroup
98 }
```

glxstr@dorecord

```
99 \newcommand*\glxstr@dorecord{%
100 \protected@write\@auxout{}\string\glxstr@record
101 {\@gls@label}\@gls@counterprefix{\@gls@counter}\@gls@numberformat}%
102 {\@gls@locref}}%
103 \glxstr@counterrecordhook
104 }
```

r@recordcounter

```
105 \newcommand*\@glxstr@recordcounter{%
106 \glxstr@noop@recordcounter
107 }
```

p@recordcounter

```
108 \newcommand*\@glxstr@noop@recordcounter}[1]{%
109 \PackageError{glossaries-extra}{\string\GlsXtrRecordCounter\space
110 requires record=only or record=alsoindex package option}{}%
111 }
```

p@recordcounter

```
112 \newcommand*\@glxstr@op@recordcounter}[1]{%
113 \eappto\glxstr@counterrecordhook{\noexpand\glxstr@docounterrecord{#1}}%
114 }
```

srtglossaryunit

```
115 \newcommand*\printunsrtglossaryunit{%
116 \print@noop@unsrtglossaryunit
117 }
```

tr@setup@record Initialise.

```
118 \newcommand*\glxstr@setup@record{}
```

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

```
119 \newcommand*\glxstr@indexonly@saveentrycounter{%
120 \ifKV@glslink@noindex
121 \else
122 \glxstr@saveentrycounter
123 \fi
124 }
```

addloclistfield

```
125 \newcommand*{\glxtr@addloclistfield}{%
126 \key@ifundefined{glossentry}{loclist}%
127 {%
128 \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
129 \appto\@gls@keymap{,{loclist}{loclist}}%
130 \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
131 \appto\@newglossaryentryposthook{%
132 \gls@assign@field{\@glo@label}{loclist}{\@glo@loclist}%
133 }%
134 \glssetnoexpandfield{loclist}%
135 }%
136 }%
```

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

```
137 \key@ifundefined{glossentry}{location}%
138 {%
139 \define@key{glossentry}{location}{\def\@glo@location{##1}}%
140 \appto\@gls@keymap{,{location}{location}}%
141 \appto\@newglossaryentryprehook{\def\@glo@location{}}%
142 \appto\@newglossaryentryposthook{%
143 \gls@assign@field{\@glo@label}{location}{\@glo@location}%
144 }%
145 \glssetnoexpandfield{location}%
146 }%
147 }%
```

Add a key to store the group heading.

```
148 \key@ifundefined{glossentry}{group}%
149 {%
150 \define@key{glossentry}{group}{\def\@glo@group{##1}}%
151 \appto\@gls@keymap{,{group}{group}}%
152 \appto\@newglossaryentryprehook{\def\@glo@group{}}%
153 \appto\@newglossaryentryposthook{%
154 \gls@assign@field{\@glo@label}{group}{\@glo@group}%
155 }%
156 \glssetnoexpandfield{group}%
157 }%
158 }%
159 }
```

Now define the record package option.

```
160 \define@choicekey{glossaries-extra.sty}{record}[\val\nr]%
161 {off,only,alsoindex}%
162 [only]%
163 {%
164 \ifcase\nr\relax
```

Don't record.

```
165 \def\glxtr@setup@record{%
```



```

166     \renewcommand*{\@glstr@record}[2]{}%
167     \let\@do@wrglossary\glstr@do@wrglossary
168     \let\@gl@saveentrycounter\glstr@indexonly@saveentrycounter
169     \let\glstrundefaction\glstr@err@undefaction
170     \let\glstr@warnonexistsordo\@gobble
171     \let\@glstr@recordcounter\glstr@noop@recordcounter
172     \def\printunsrtglossaryunit{\print@noop@unsrtglossaryunit}%
173     \undef\glstrsetaliasnoindex
174 }%
175 \or

```

Only record (don't index).

```

176 \def\glstr@setup@record{%
177   \let\@glstr@record\@glstr@record
178   \let\@do@wrglossary\@gobble
179   \let\@gl@saveentrycounter\relax
180   \let\glstrundefaction\glstr@warn@undefaction
181   \let\glstr@warnonexistsordo\glstr@warn@onexistsordo
182   \glstr@addloclistfield
183   \renewcommand*{\@glstr@autoindexcrossrefs}{}%
184   \let\@glstr@recordcounter\glstr@op@recordcounter
185   \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%

```

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

```

186   \def\glstrsetaliasnoindex{%
187   }%
188 \or

```

Record and index.

```

189 \def\glstr@setup@record{%
190   \let\@glstr@record\@glstr@record
191   \let\@do@wrglossary\glstr@do@wrglossary
192   \let\@gl@saveentrycounter\glstr@indexonly@saveentrycounter
193   \let\glstrundefaction\glstr@warn@undefaction
194   \let\glstr@warnonexistsordo\glstr@warn@onexistsordo
195   \glstr@addloclistfield
196   \let\@glstr@recordcounter\glstr@op@recordcounter
197   \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
198   \undef\glstrsetaliasnoindex
199 }%
200 \fi
201 }

```

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.

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

```
202 \newcount\@glstr@docdefval
```

Need to provide conditional commands that are backward compatible:

if@glxtrdocdef

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

lsxtrdocdeftrue

```
204 \newcommand*{\@glxtrdocdeftrue}{\@glxtr@docdefval=1 }
```

lsxtrdocdeffalse

```
205 \newcommand*{\@glxtrdocdeffalse}{\@glxtr@docdefval=0 }
```

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

```
206 \define@choicekey{glossaries-extra.sty}{docdef}[\val\nr]%
```

```
207 {false,true,restricted}[true]%
```

```
208 {%
```

```
209   \@glxtr@docdefval=\nr\relax
```

```
210   \ifnum\@glxtr@docdefval=2\relax
```

```
211     \renewcommand*{\@glsdofexistsorwarn}{\glsdofexists}%
```

```
212   \fi
```

```
213 }
```

ocdefrestricted

```
214 \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).

```
215 \newcommand*{\@glsdofexistsorwarn}{\glsdofexistsorwarn}
```

indexcrossrefs

Automatically index cross references at the end of the document

```
216 \define@boolkey{glossaries-extra.sty}[@glxtr]{indexcrossrefs}[true]{%
```

```
217   \if@glxtrindexcrossrefs
```

```
218   \else
```

```
219     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
```

```
220   \fi
```

```
221 }
```

Switch off since this can increase the build time.

```
222 \@glxtrindexcrossrefsfalse
```

But allow see key to switch it on automatically.

oindexcrossrefs

```
223 \newcommand*{\@glxtr@autoindexcrossrefs}{\@glxtrindexcrossrefstrue}
```

iesExtraWarning

Allow users to suppress warnings.

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

```

raWarningNoLine  Allow users to suppress warnings.
225 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
226 \PackageWarningNoLine{glossaries-extra}{#1}}

227 \@glsxtr@declareoption{nowarn}{%
228 \let\GlossariesExtraWarning\@gobble
229 \let\GlossariesExtraWarningNoLine\@gobble
230 \glsxtr@doption{nowarn}%
231 }

postdot  Shortcut for nopostdot=false
232 \@glsxtr@declareoption{postdot}{%
233 \glsxtr@doption{nopostdot=false}%
234 }

glsxtrabbrvtype  Glossary type for abbreviations.
235 \newcommand*{\glsxtrabbrvtype}{\glsdefaulttype}

abbreviationsdef  Set by abbreviations option.
236 \newcommand*{\@glsxtr@abbreviationsdef}{}

abbreviationsdef
237 \newcommand*{\@glsxtr@doabbreviationsdef}{%
238 \ifpackageloaded{babel}%
239 {\providecommand{\abbreviationsname}{\acronymname}}%
240 {\providecommand{\abbreviationsname}{Abbreviations}}%
241 \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
242 \renewcommand*{\glsxtrabbrvtype}{abbreviations}%
243 \newcommand*{\printabbreviations}[1][1]{%
244 \printglossary[type=\glsxtrabbrvtype,##1]%
245 }%
246 \disable@keys{glossaries-extra.sty}{abbreviations}%

  If the acronym option hasn't been used, change \acronymtype to \glsxtrabbrvtype.
247 \ifglsacronym
248 \else
249 \renewcommand*{\acronymtype}{\glsxtrabbrvtype}%
250 \fi
251 }%

abbreviations  If abbreviations, create a new glossary type for abbreviations.
252 \@glsxtr@declareoption{abbreviations}{%
253 \let\@glsxtr@abbreviationsdef\@glsxtr@doabbreviationsdef
254 }

iationShortcuts  Enable shortcut commands for the abbreviations. Unlike the analogous command provided
  by glossaries, this uses \newcommand instead of \let as a safety feature.
255 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
256 \newcommand*{\ab}{\cgl}%

```

```

257 \newcommand*{\abp}{\cglsp1}%
258 \newcommand*{\as}{\glxtrshort}%
259 \newcommand*{\asp}{\glxtrshortpl}%
260 \newcommand*{\al}{\glxtrlong}%
261 \newcommand*{\alp}{\glxtrlongpl}%
262 \newcommand*{\af}{\glxtrfull}%
263 \newcommand*{\afp}{\glxtrfullpl}%
264 \newcommand*{\Ab}{\cGls}%
265 \newcommand*{\Abp}{\cGlspl}%
266 \newcommand*{\As}{\Glsxtrshort}%
267 \newcommand*{\Asp}{\Glsxtrshortpl}%
268 \newcommand*{\Al}{\Glsxtrlong}%
269 \newcommand*{\Alp}{\Glsxtrlongpl}%
270 \newcommand*{\Af}{\Glsxtrfull}%
271 \newcommand*{\Afp}{\Glsxtrfullpl}%
272 \newcommand*{\AB}{\cGLS}%
273 \newcommand*{\ABP}{\cGLSpl}%
274 \newcommand*{\AS}{\GLSxtrshort}%
275 \newcommand*{\ASP}{\GLSxtrshortpl}%
276 \newcommand*{\AL}{\GLSxtrlong}%
277 \newcommand*{\ALP}{\GLSxtrlongpl}%
278 \newcommand*{\AF}{\GLSxtrfull}%
279 \newcommand*{\AFP}{\GLSxtrfullpl}%
280 \newcommand*{\newabbr}{\newabbreviation}%

```

Disable this command after it's been used.

```

281 \let\GlsXtrDefineAbbreviationShortcuts\relax
282 }

```

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

```

283 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
284 \newcommand*{\newentry}{\newglossaryentry}%
285 \ifdef\printsymbols
286 {%
287 \newcommand*{\newsym}{\glxtrnewsymbol}%
288 }{}%
289 \ifdef\printnumbers
290 {%
291 \newcommand*{\newnum}{\glxtrnewnumber}%
292 }{}%
293 \let\GlsXtrDefineOtherShortcuts\relax
294 }

```

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.

```

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

```

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

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

Provide shortcuts option. Unlike the glossaries version, this is a choice rather than a boolean key but it also provides shortcuts=true and shortcuts=false, which are equivalent to shortcuts=all and shortcuts=none. Multiple use of this option in the *same* option list will override each other.

```
297 \define@choicekey{glossaries-extra.sty}{shortcuts}[\val\nr]%
298 {acronyms,acro,abbreviations,abbr,other,all,true,none,false}[true]{%
299   \let\@glxtr@shortcutsval\val
300   \ifcase\nr\relax % acronyms
301     \renewcommand*{\@glxtr@setupshortcuts}{%
302       \glacrshortcutstrue
303       \DefineAcronymSynonyms
304     }%
305   \or % acro
306     \renewcommand*{\@glxtr@setupshortcuts}{%
307       \glacrshortcutstrue
308       \DefineAcronymSynonyms
309     }%
310   \or % abbreviations
311     \renewcommand*{\@glxtr@setupshortcuts}{%
312       \GlsXtrDefineAbbreviationShortcuts
313     }%
314   \or % abbr
315     \renewcommand*{\@glxtr@setupshortcuts}{%
316       \GlsXtrDefineAbbreviationShortcuts
317     }%
318   \or % other
319     \renewcommand*{\@glxtr@setupshortcuts}{%
320       \GlsXtrDefineOtherShortcuts
321     }%
322   \or % all
323     \renewcommand*{\@glxtr@setupshortcuts}{%
324       \glacrshortcutstrue
325       \DefineAcronymSynonyms
326       \GlsXtrDefineAbbreviationShortcuts
327       \GlsXtrDefineOtherShortcuts
328     }%
329   \or % true
330     \renewcommand*{\@glxtr@setupshortcuts}{%
331       \glacrshortcutstrue
332       \DefineAcronymSynonyms
333       \GlsXtrDefineAbbreviationShortcuts
334       \GlsXtrDefineOtherShortcuts
335     }%
336   \else % none, false
337     \renewcommand*{\@glxtr@setupshortcuts}{}%
338   \fi
```

```

339 }

\lsxtr@doaccsupp
340 \newcommand*{\@glxtr@doaccsupp}{%

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

GlossaryWarning Warning text displayed in document if the external glossary file given by the argument is miss-
ing.
343 \newcommand{\glxtrNoGlossaryWarning}[1]{%
344 \@glxtr@defaultnoglossarywarning{#1}%
345 }

nomissingglsstext If true, suppress the text produced if the external glossary file is missing.
346 \define@choicekey{glossaries-extra.sty}{nomissingglsstext}[\val\nr]%
347 {true,false}[true]{%
348 \ifcase\nr\relax % true
349 \renewcommand{\glxtrNoGlossaryWarning}[1]{%
350 \null
351 }%
352 \else % false
353 \renewcommand{\glxtrNoGlossaryWarning}[1]{%
354 \@glxtr@defaultnoglossarywarning{#1}%
355 }%
356 \fi
357 }

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

\lsxtr@redefstyles
358 \newcommand*{\@glxtr@redefstyles}{%

stylemods
359 \define@key{glossaries-extra.sty}{stylemods}{%
360 \ifblank{#1}%
361 {%
362 \renewcommand*{\@glxtr@redefstyles}{%
363 \RequirePackage{glossaries-extra-stylemods}}%
364 }%
365 {%
366 \renewcommand*{\@glxtr@redefstyles}{}%
367 \@for\@glxtr@tmp:=#1\do{%
368 \IfFileExists{glossary-\@glxtr@tmp.sty}%
369 {%
370 \eappto\@glxtr@redefstyles{%
371 \noexpand\RequirePackage{glossary-\@glxtr@tmp}}%

```

```

372     }%
373     {%
374         \PackageError{glossaries-extra}%
375         {Glossaries style package ‘glossary- \@glsxtr@tmp.sty’
376         doesn’t exist (did you mean to use the ‘style’ key?)}%
377         {The list of values (#1) in the ‘stylemods’ key should
378         match the glossary-xxx.sty files provided with
379         glossaries.sty}%
380     }%
381 }%
382 \appto\@glsxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
383 }%
384 }

```

glsxtr@do@style

```

385 \newcommand*{\@glsxtr@do@style}{ }

```

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

```

386 \define@key{glossaries-extra.sty}{style}{%
387 \renewcommand*{\@glsxtr@do@style}{%

```

Set this as the default style:

```

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

```

Set this style:

```

389 \setglossarystyle{#1}%
390 }%
391 }

```

Pass all other options to glossaries.

```

392 \DeclareOptionX*{%
393 \expandafter\glsxtr@doooption\expandafter{\CurrentOption}}

```

Process options.

```

394 \ProcessOptionsX

```

Load glossaries if not already loaded.

```

395 \RequirePackage{glossaries}

```

Load the glossaries-accsupp package if required.

```

396 \@glsxtr@doaccsupp

```

Define abbreviations glossaries if required.

```

397 \@glsxtr@abbreviationsdef
398 \let\@glsxtr@abbreviationsdef\relax

```

Setup shortcuts if required.

```

399 \@glsxtr@setupshortcuts

```

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

```

400 \@glsxtr@redef@for@gl@sentries

```

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

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

Now define the user command:

```
402 \newcommand*\glossariesextrasetup[1]{%
403   \let\glxtr@setup@record\relax
404   \let\glxtr@setup@shortcuts\relax
405   \let\glxtr@redef@for@gl@sentries\relax
406   \setkeys{glossaries-extra.sty}{#1}%
407   \glxtr@abbreviationsdef
408   \let\glxtr@abbreviationsdef\relax
409   \glxtr@setup@shortcuts
410   \glxtr@setup@record
411   \glxtr@redef@for@gl@sentries
412 }
```

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

```
413 \let\glxtr@do@wrglossary\do@wrglossary
```

aveentrycounter Save original definition of `\@gl@s@saveentrycounter`.

```
414 \let\glxtr@s@saveentrycounter\@gl@s@saveentrycounter
```

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

```
415 \let\@gl@s@saveentrycounter\glxtr@indexonly@s@saveentrycounter
```

Set up record option if required.

```
416 \glxtr@setup@record
```

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

```
417 \AtBeginDocument{%
418   \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
419   \def\glxtrundef@tag{\glxtrundef@tag}%
420 }
```

1.2 Extra Utilities

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

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

has been defined and is simply a comma, the glossary exists and is empty. (It's initialised to a comma.)

```

421 \newcommand{\glxtrifemptyglossary}[3]{%
422   \ifcsdef{glolist@#1}%
423   {%
424     \ifcsstring{glolist@#1}{,}{#2}{#3}%
425   }%
426   {%
427     \glxtrundefaction{Glossary type '#1' doesn't exist}{}%
428     #2%
429   }%
430 }
```

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

```

431 \newcommand*{\glxtrifkeydefined}[3]{%
432   \key@ifundefined{glossentry}{#1}{#3}{#2}%
433 }
```

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

```

434 \newcommand*{\glxtrprovidestoragekey}{%
435   \@ifstar\@sglsxtr@provide@storagekey\@glxtr@provide@storagekey
436 }
```

vide@storagekey Unstarred version.

```

437 \newcommand*{\@glxtr@provide@storagekey}[3]{%
438   \key@ifundefined{glossentry}{#1}%
439   {%
440     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
441     \appto\@gl s@keymap{,}{#1}{#1}}%
442     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
443     \appto\@newglossaryentryposthook{%
444       \letcs{\@glo@tmp}{@glo@#1}%
445       \gl s@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
446     }%
```

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

```

447   \ifblank{#3}
448   {%
449   {%
450     \newcommand*{#3}[1]{\@gl s@entry@field{##1}{#1}}%
451   }%
452   }%
453   {%
```

Provide the no-link command if not already defined.

```

454   \ifblank{#3}
455   {%
456   {%
```

```

457     \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
458   }%
459 }%
460 }

```

`\provide@storagekey` Starred version.

```

461 \newcommand*\s@glstr@provide@storagekey}[1]{%
462   \key@ifundefined{glossentry}{#1}%
463   {%
464     \expandafter\newcommand\expandafter*\expandafter
465     {\csname gls@assign@#1@field\endcsname}[2]{%
466       \@gls@expand@field{##1}{#1}{##2}%
467     }%
468   }%
469   {}%
470   \@glstr@provide@addstoragekey{#1}%
471 }

```

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

`\GlsXtrFmtField`

```

472 \newcommand{\GlsXtrFmtField}{useri}

```

`\GlsXtrFmtDefaultOptions`

```

473 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

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

```

474 \newrobustcmd*\glstrfmt[3][[]]{%
475   \glsdoifexistsordo{#2}%
476   {%
477     \ifglshasfield{\GlsXtrFmtField}{#2}%
478     {%
479       \let\do@gl@link@checkfirsthyper\relax
480       \expandafter\@gl@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
481       {\csuse{\glscurrentfieldvalue}{#3}}%
482     }%
483     {#3}%
484   }%
485   {#3}%
486 }

```

`\glxtrentryfmt` No link or indexing.

```

487 \ifdef\texorpdfstring
488 {
489   \newcommand*\glxtrentryfmt[2]{%

```

```

490 \texorpdfstring{\@glstrententryfmt{#1}{#2}}{#2}%
491 }
492 }
493 {
494 \newcommand*{\glstrententryfmt}{\@glstrententryfmt}
495 }

```

@glstrententryfmt

```

496 \newrobustcmd*{\@glstrententryfmt}[2]{%
497 \glsoifexistsordo
498 {%
499 \ifglshasfield{\GlsXtrFmtField}{#1}%
500 {%
501 \csuse{\glscurrentfieldvalue}{#2}%
502 }%
503 {#2}%
504 }%
505 {#2}%
506 }

```

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

```

507 \newcommand*{\glxtrfieldlistadd}[3]{%
508 \listcsadd{glo@\glsetoklabel{#1}@#2}{#3}%
509 }

```

trfieldlistgadd Similarly but uses \listcsgadd.

```

510 \newcommand*{\glxtrfieldlistgadd}[3]{%
511 \listcsgadd{glo@\glsetoklabel{#1}@#2}{#3}%
512 }

```

trfieldlisteadadd Similarly but uses \listcseadd.

```

513 \newcommand*{\glxtrfieldlisteadadd}[3]{%
514 \listcseadd{glo@\glsetoklabel{#1}@#2}{#3}%
515 }

```

trfieldlistxadd Similarly but uses \listcsxadd.

```

516 \newcommand*{\glxtrfieldlistxadd}[3]{%
517 \listcsxadd{glo@\glsetoklabel{#1}@#2}{#3}%
518 }

```

Now provide commands to iterate over these lists.

fielddolistloop

```

519 \newcommand*{\glxtrfielddolistloop}[2]{%
520 \dolistcsloop{glo@\glsetoklabel{#1}@#2}%
521 }

```

ieldforlistloop

```
522 \newcommand*{\glxtrfieldforlistloop}[3]{%
523   \forlistcsloop{glo@\glsdetoklabel{#1}@#2}{#3}%
524 }
```

List element tests:

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

```
525 \newcommand*{\glxtrfieldifinlist}[5]{%
526   \ifinlistcs{#3}{glo@\glsdetoklabel{#1}@#2}{#4}{#5}%
527 }
```

rfieldxifinlist Expands item.

```
528 \newcommand*{\glxtrfieldxifinlist}[5]{%
529   \xifinlistcs{#3}{glo@\glsdetoklabel{#1}@#2}{#4}{#5}%
530 }
```

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

```
531 \newcommand*{\glxtrusefield}[2]{%
532   \@gls@entry@field{#1}{#2}%
533 }
```

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

```
534 \newcommand*{\Glsxtrusefield}[2]{%
535   \@gls@entry@field{#1}{#2}%
536 }
```

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

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

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

```
538 \newcommand*{\glxstredeffield}[2]{\csedef{glo@\glsdetoklabel{#1}@#2}{}}
```

etfieldifexists

```
539 \newcommand*{\glxtrsetfieldifexists}[3]{\glsoifexists{#1}{#3}}
```

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

```
540 \newrobustcmd*{\GlsXtrSetField}[3]{%
541   \glxtrsetfieldifexists{#1}{#2}%
542   {\csdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
543 }
```

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

```

544 \newrobustcmd*{\GlsXtrLetField}[3]{%
545   \glsxtrsetfieldifexists{#1}{#2}%
546   {\cslet{glo@\glsdetoklabel{#1}@#2}{#3}}%
547 }

```

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

```

548 \newrobustcmd*{\csGlsXtrLetField}[3]{%
549   \glsxtrsetfieldifexists{#1}{#2}%
550   {\csletcs{glo@\glsdetoklabel{#1}@#2}{#3}}%
551 }

```

`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.

```

552 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
553   \glsxtrsetfieldifexists{#1}{#2}%
554   {\csletcs{glo@\glsdetoklabel{#1}@#2}{glo@\glsdetoklabel{#3}@#4}}%
555 }

```

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

```

556 \newrobustcmd*{\gGlsXtrSetField}[3]{%
557   \glsxtrsetfieldifexists{#1}{#2}%
558   {\csgdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
559 }

```

`xGlsXtrSetField`

```

560 \newrobustcmd*{\xGlsXtrSetField}[3]{%
561   \glsxtrsetfieldifexists{#1}{#2}%
562   {\protected@csxdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
563 }

```

`eGlsXtrSetField`

```

564 \newrobustcmd*{\eGlsXtrSetField}[3]{%
565   \glsxtrsetfieldifexists{#1}{#2}%
566   {\protected@csedef{glo@\glsdetoklabel{#1}@#2}{#3}}%
567 }

```

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

```

568 \ifglsentrycounter
569   \newcommand*{\glsxtrpageref}[1]{\pageref{glsentry-\glsdetoklabel{#1}}}
570 \else
571   \ifglsesubentrycounter
572     \newcommand*{\glsxtrpageref}[1]{\pageref{glsentry-\glsdetoklabel{#1}}}
573   \else
574     \newcommand*{\glsxtrpageref}[1]{\gls{#1}}
575   \fi
576 \fi

```

lossarypreamble

```
577 \newcommand{\apptoglossarypreamble}[2][\glsdefaulttype]{%
578   \ifcsdef{glolist@#1}%
579   {%
580     \ifcsundef{@glossarypreamble@#1}%
581     {\csdef{@glossarypreamble@#1}{}}%
582     {}%
583     \csappto{@glossarypreamble@#1}{#2}%
584   }%
585   {%
586     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
587   }%
588 }
```

lossarypreamble

```
589 \newcommand{\preglossarypreamble}[2][\glsdefaulttype]{%
590   \ifcsdef{glolist@#1}%
591   {%
592     \ifcsundef{@glossarypreamble@#1}%
593     {\csdef{@glossarypreamble@#1}{}}%
594     {}%
595     \cspreteto{@glossarypreamble@#1}{#2}%
596   }%
597   {%
598     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
599   }%
600 }
```

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.

\glsxtralias Provide a key to allow aliases to be defined. The key should be set to the label of the synonymous entry.

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

ryentryposthook Append to the hook to check for the alias key.

```
602 \appto\@newglossaryentryposthook{%
603   \ifcsvoid{glo@\glo@label @alias}{}%
604   {%
605     \ifdefvoid\@glo@see
606     {%
607       \edef\@do@glsee{\noexpand\glsee
608         {\@glo@label}{\csuse{glo@\glo@label @alias}}}%
609       \@do@glsee
```

```

610 }%
611 {}%
612 }%
613 }

```

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

`ewglossaryentry`

```

614 \renewcommand*{\longnewglossaryentry}{%
615 \ifstar\@glstr@s@longnewglossaryentry\@glstr@longnewglossaryentry
616 }

```

`ewglossaryentry`

Starred version.

```

617 \newcommand{\@glstr@s@longnewglossaryentry}[3]{%
618 \glsdoifnoexists{#1}%
619 {%
620 \bgroup
621 \let\@org@newglossaryentryprehook\@newglossaryentryprehook
622 \long\def\@newglossaryentryprehook{%
623 \long\def\@glo@desc{#3}%
624 \@org@newglossaryentryprehook
625 }%
626 \renewcommand*{\gls@assign@desc}[1]{%
627 \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
628 \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
629 }
630 \gls@defglossaryentry{#1}{#2}%
631 \egroup
632 }%
633 }

```

`ewglossaryentry`

Unstarred version.

```

634 \newcommand{\@glstr@longnewglossaryentry}[3]{%
635 \glsdoifnoexists{#1}%
636 {%
637 \bgroup
638 \let\@org@newglossaryentryprehook\@newglossaryentryprehook
639 \long\def\@newglossaryentryprehook{%
640 \long\def\@glo@desc{#3\glstrpostlongdescription}%
641 \@org@newglossaryentryprehook
642 }%
643 \renewcommand*{\gls@assign@desc}[1]{%
644 \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%

```

The following is different from the base glossaries.sty:

```

645 \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
646 }

```

```

647     \gls@defglossaryentry{#1}{#2}%
648   \egroup
649 }%
650 }

```

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

```

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

```

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

`\ignoredglossary` Redefine to check for star.

```

652 \renewcommand{\newignoredglossary}{%
653   \@ifstar\glsxtr@s@newignoredglossary\glsxtr@org@newignoredglossary
654 }

```

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

```

655 \newcommand*{\glsxtr@org@newignoredglossary}[1]{%
656   \ifcsdef{glolist@#1}
657   {%
658     \glsxtrundefaction{Glossary type ‘#1’ already exists}{}%
659   }%
660   {%
661     \ifdefempty\@ignored@glossaries
662     {%
663       \edef\@ignored@glossaries{#1}%
664     }%
665     {%
666       \eappto\@ignored@glossaries{,#1}%
667     }%
668     \csgdef{glolist@#1}{,}%
669     \ifcsundef{gls@#1@entryfmt}%
670     {%
671       \defglsentryfmt[#1]{\glsentryfmt}%
672     }%
673     {}%
674     \ifdefempty\@gls@nohyperlist
675     {%
676       \renewcommand*{\@gls@nohyperlist}{#1}%
677     }%
678     {%
679       \eappto\@gls@nohyperlist{,#1}%
680     }%
681   }%
682 }

```

`\ignoredglossary` Starred form.

```

683 \newcommand*{\glsxtr@s@newignoredglossary}[1]{%
684   \ifcsdef{glolist@#1}

```



```

685 {%
686   \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
687 }%
688 {%
689   \ifdefempty\@ignored@glossaries
690   {%
691     \edef\@ignored@glossaries{#1}%
692   }%
693   {%
694     \eappto\@ignored@glossaries{,#1}%
695   }%
696   \csgdef{glolist@#1}{,}%
697   \ifcsundef{gls@#1@entryfmt}%
698   {%
699     \defglsentryfmt[#1]{\glsentryfmt}%
700   }%
701   {}%
702 }%
703 }

```

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

```

704 \glsifusetranslator
705 {%
706   \renewcommand*{\glsettoctitle}[1]{%
707     \ifcsdef{gls@tr@set@#1@toctitle}%
708     {%
709       \csuse{gls@tr@set@#1@toctitle}%
710     }%
711     {%
712       \ifcsdef{@glotype@#1@title}%
713       {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
714       {\def\glossarytoctitle{\glossarytitle}}%
715     }%
716   }%
717 }
718 {
719   \renewcommand*{\glsettoctitle}[1]{%
720     \ifcsdef{@glotype@#1@title}%
721     {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
722     {\def\glossarytoctitle{\glossarytitle}}%
723   }
724 }

```

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

```

725 \newcommand{\provideignoredglossary}{%
726   \@ifstar\glxtr@s@provideignoredglossary\glxtr@provideignoredglossary
727 }

```

ignoredglossary Unstarred version.

```
728 \newcommand*{\glxtr@provideignoredglossary}[1]{%
729   \ifcsdef{glolist@#1}
730   {}%
731   {%
732     \ifdefempty\@ignored@glossaries
733     {%
734       \edef\@ignored@glossaries{#1}%
735     }%
736     {%
737       \eappto\@ignored@glossaries{, #1}%
738     }%
739     \csgdef{glolist@#1}{,}%
740     \ifcsundef{gls@#1@entryfmt}%
741     {%
742       \defglsentryfmt[#1]{\glsentryfmt}%
743     }%
744     {}%
745     \ifdefempty\@gls@nohyperlist
746     {%
747       \renewcommand*{\@gls@nohyperlist}{#1}%
748     }%
749     {%
750       \eappto\@gls@nohyperlist{, #1}%
751     }%
752   }%
753 }
```

ignoredglossary Starred form.

```
754 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
755   \ifcsdef{glolist@#1}
756   {}%
757   {%
758     \ifdefempty\@ignored@glossaries
759     {%
760       \edef\@ignored@glossaries{#1}%
761     }%
762     {%
763       \eappto\@ignored@glossaries{, #1}%
764     }%
765     \csgdef{glolist@#1}{,}%
766     \ifcsundef{gls@#1@entryfmt}%
767     {%
768       \defglsentryfmt[#1]{\glsentryfmt}%
769     }%
770     {}%
771   }%
772 }
```

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

```

773 \newcommand*{\glxtrcopytoglossary}[2]{%
774   \glsdoidexists{#1}%
775   {%
776     \ifcsdef{glolist@#2}
777     {%
778       \cseappto{glolist@#2}{#1,}%
779     }%
780     {%
781       \glxtrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
782     }%
783   }%
784 }

```

1.3.1 Existence Checks

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

```

785 \renewcommand{\glsdoidexists}[2]{%
786   \ifglstryexists{#1}{#2}%
787   {%

```

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

```

788   \edef\glslabel{\glsdetoklabel{#1}}%
789   \glxtrundefaction{Glossary entry ‘\glslabel’
790     has not been defined}{You need to define a glossary entry before
791     you can reference it.}%
792   }%
793 }

```

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

```

794 \renewcommand{\glsdoidnoexists}[2]{%
795   \ifglstryexists{#1}{%
796     \glxtrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
797       has already been defined}{}}{#2}%
798 }

```

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

```

799 \ifdef\glsdoidexistsordo
800 {%
801   \renewcommand{\glsdoidexistsordo}[3]{%
802     \ifglstryexists{#1}{#2}%
803     {%
804       \glxtrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
805         has not been defined}{You need to define a glossary entry
806         before you can use it.}%

```

```

807      #3%
808    }%
809  }%
810 }
811 {%
812   \glstr@warnonexistsordo\glsdoifexistsordo
813   \newcommand{\glsdoifexistsordo}[3]{%
814     \ifglstryexists{#1}{#2}%
815     {%
816       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
817       has not been defined}{You need to define a glossary entry
818       before you can use it.}%
819     #3%
820     }%
821   }%
822 }

```

arynoexistsordo Similarly for \doifglossarynoexistsordo.

```

823 \ifdef\doifglossarynoexistsordo
824 {%
825   \renewcommand{\doifglossarynoexistsordo}[3]{%
826     \ifglossaryexists{#1}%
827     {%
828       \glstrundefaction{Glossary type '#1' already exists}{}%
829     #3%
830     }%
831     {#2}%
832   }%
833 }
834 {%
835   \glstr@warnonexistsordo\doifglossarynoexistsordo
836   \newcommand{\doifglossarynoexistsordo}[3]{%
837     \ifglossaryexists{#1}%
838     {%
839       \glstrundefaction{Glossary type '#1' already exists}{}%
840     #3%
841     }%
842     {#2}%
843   }%
844 }
845

```

ryentryposthook Hook into end of \newglossaryentry to add “see” value as a field.

```

846 \appto\@newglossaryentryposthook{%
847   \ifdefvoid\@glo@see
848   {\csxdef{glo@\@glo@label @see}{}}%
849   {%
850     \csxdef{glo@\@glo@label @see}{\@glo@see}%
851     \@glstr@autoindexcrossrefs

```

```

852 }%
853 }
854 \appto\@gls@keymap{,{see}{see}}

```

`\glstrusee` Apply `\glseeformat` to the see key if not empty.

```

855 \newcommand*{\glstrusee}[1]{%
856   \glsoifexists{#1}%
857   {%
858     \letcs{\@glo@see}{glo@\glsetoklabel{#1}@see}%
859     \ifdefempty\@glo@see
860     {%
861       {%
862         \expandafter\glstr@usee\@glo@see\@end@glstr@usee
863       }%
864     }%
865 }

```

`\glstr@usee`

```

866 \newcommand*{\glstr@usee}[1][\seename]{%
867   \@glstr@usee[1]%
868 }

```

`\@glstr@usee`

```

869 \def\@glstr@usee[#1]#2\@end@glstr@usee{%
870   \glstruseeformat{#1}{#2}%
871 }

```

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

```

872 \newcommand*{\glstruseeformat}[2]{%
873   \glseeformat{#1}{#2}{}%
874 }

```

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

```

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

```

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

```

876 \newcommand*{\glstraddallcrossrefs}{%
877   \forallglossaries{\@glo@type}%
878   {%
879     \forallglsentries[\@glo@type]{\@glo@label}%
880     {%
881       \ifglused{\@glo@label}{\@glstr@addunusedxrefs{\@glo@label}}{}}%
882     }%
883   }%
884 }

```

@addunusedxrefs If the given entry has a see field add all unused cross-references.

```
885 \newcommand*{\@glsxtr@addunusedxrefs}[1]{%
886   \letcs{\@glo@see}{glo\glsdetoklabel{#1}@see}%
887   \ifdefvoid\@glo@see
888   {}%
889   {%
890     \expandafter\glsxtr@addunused\@glo@see\@end@glsxtr@addunused
891   }%
892 }
```

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

```
893 \newcommand*{\glsxtr@addunused}[1][]{%
894   \@glsxtr@addunused
895 }
```

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

```
896 \def\@glsxtr@addunused#1\@end@glsxtr@addunused{%
897   \@for\@glsxtr@label:=#1\do
898   {%
899     \ifglsused{\@glsxtr@label}{}%
900     {%
901       \glsadd[format=glsxtrunusedformat]{\@glsxtr@label}%
902       \glsunset{\@glsxtr@label}%
903       \@glsxtr@addunusedxrefs{\@glsxtr@label}%
904     }%
905   }%
906 }
```

xtrunusedformat

```
907 \newcommand*{\glsxtrunusedformat}[1]{\unskip}
```

1.3.2 Document Definitions

noidxglossaries Modify \makenoidxglossaries so that it automatically switches off (unless the restricted setting is on) and disables the docdef key.

```
908 \let\glsxtr@orgmakenoidxglossaries\makenoidxglossaries
909 \renewcommand{\makenoidxglossaries}{%
910   \glsxtr@orgmakenoidxglossaries
911   \if@glsxtrdocdefrestricted
```

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

```
912   \renewcommand*{\@gls@reference}[3]{%
913     \ifcsundef{@glsref@##1}{\csgdef{@glsref@##1}{}}{}%
914     \ifinlistcs{##2}{@glsref@##1}%
915     {}%
916     {\listcsgadd{@glsref@##1}{##2}}%
917     \ifcsundef{glo\glsdetoklabel{##2}@loclist}%

```

```

918      {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
919      }}%
920      \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
921      }%
922  \else

  Disable document definitions.
923      \@glxtrdocdeffalse
924  \fi
925  \disable@keys{glossaries-extra.sty}{docdef}%
926 }

```

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

```

927 \renewcommand*{\gls@defdocnewglossaryentry}{%
928   \ifcase\@glxtr@docdefval
     docdef=false:
929     \renewcommand*{\newglossaryentry}[2]{%
930       \PackageError{glossaries-extra}{Glossary entries must
931         be \MessageBreak defined in the preamble with \MessageBreak
932         package option ‘docdef=false’\MessageBreak(consider using
933         ‘docdef=restricted’)}{Move your glossary definitions to
934         the preamble. You can also put them in a \MessageBreak separate file
935         and load them with \string\loadglsentries.}%
936     }%
937   \or

     docdef=true Since the see value is now saved in a field, it can be used by entries that have
     been defined in the document.
938     \let\gls@checkseeallowed\relax
939     \let\newglossaryentry\new@glossaryentry
940   \or

     Restricted mode just needs to allow the see value.
941     \let\gls@checkseeallowed\relax
942   \fi
943 }%

```

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`

```

944 \newcommand*{\GlsXtrEnableOnTheFly}{%
945   \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
946 }

```

`\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.)

```

947 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
948   \renewcommand*{\glsdetoklabel}[1]{%
949     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
950     {%
951       \expandafter\detokenize\expandafter{##1}%
952     }%
953     {\detokenize{##1}}}%
954   }%
955   \@GlsXtrEnableOnTheFly
956 }
957 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
958   \expandafter\if\glsbackslash#1%
959     #3%
960   \else
961     #4%
962   \fi
963 }

```

sxtrstarflywarn

```

964 \newcommand*{\glsxtrstarflywarn}{%
965   \GlossariesExtraWarning{Experimental starred version of
966   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
967   read the warnings in the glossaries-extra user manual)}}%
968 }

```

rEnableOnTheFly

```

969 \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`

```

970 \newcommand*{\glsxtrcat}{general}

```

`\glsxtr`

```

971 \newcommand*{\glsxtr}[1] []{%
972   \def\glsxtr@keylist{##1}%
973   \@glsxtr
974 }

```

`\@glsxtr`

```

975 \newcommand*{\@glsxtr}[2] []{%
976   \ifglentryexists{##2}%
977   {%
978     \ifblank{##1}{\@GlsXtrWarning{##1}{##2}}%

```



```

979 }%
980 {%
981   \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
982     description={\nopostdesc},##1}%
983 }%
984 \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
985 }

```

\Glsxtr

```

986 \newcommand*\Glsxtr}[1][]{%
987   \def\glsxtr@keylist{##1}%
988   \@Glsxtr
989 }

```

\@Glsxtr

```

990 \newcommand*\@Glsxtr}[2][]{%
991   \ifglsentryexists{##2}%
992   {%
993     \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
994   }%
995   {%
996     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
997       description={\nopostdesc},##1}%
998   }%
999   \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
1000 }

```

\glsxtrpl

```

1001 \newcommand*\glsxtrpl}[1][]{%
1002   \def\glsxtr@keylist{##1}%
1003   \@glsxtrpl
1004 }

```

\@glsxtrpl

```

1005 \newcommand*\@glsxtrpl}[2][]{%
1006   \ifglsentryexists{##2}%
1007   {%
1008     \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
1009   }%
1010   {%
1011     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1012       description={\nopostdesc},##1}%
1013   }%
1014   \expandafter\glspl\expandafter[\glsxtr@keylist]{##2}%
1015 }

```

\Glsxtrpl

```

1016 \newcommand*\Glsxtrpl}[1][]{%
1017   \def\glsxtr@keylist{##1}%

```

```

1018 \Glsxtrpl
1019 }

```

\Glsxtrpl

```

1020 \newcommand*{\Glsxtrpl}[2][]{%
1021 \ifglstryexists{##2}
1022 {%
1023 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
1024 }%
1025 {%
1026 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1027 description={\nopostdesc},##1}%
1028 }%
1029 \expandafter\Glspl\expandafter[\glstr@keylist]{##2}%
1030 }

```

\GlsXtrWarning

```

1031 \newcommand*{\GlsXtrWarning}[2]{%
1032 \def\glstr@optlist{##1}%
1033 \@onelevel@sanitize\glstr@optlist
1034 \GlossariesExtraWarning{The options '\glstr@optlist' have
1035 been ignored for entry '##2' as it has already been defined}%
1036 }

```

Disable commands after the glossary:

```

1037 \renewcommand\@printglossary[2]{%
1038 \def\glstr@printglossopts{##1}%
1039 \@glstr@orgprintglossary{##1}{##2}%
1040 \def\glstr{\glstr@disabledflycommand\glstr}%
1041 \def\glstrpl{\glstr@disabledflycommand\glstrpl}%
1042 \def\Glsxtr{\glstr@disabledflycommand\Glsxtr}%
1043 \def\Glsxtrpl{\glstr@disabledflycommand\Glsxtrpl}%
1044 }

```

abledflycommand

```

1045 \newcommand*{\glstr@disabledflycommand}[1]{%
1046 \PackageError{glossaries-extra}%
1047 {\string##1\space can't be used after any of the \MessageBreak
1048 glossaries have been displayed}%
1049 {The on-the-fly commands enabled by
1050 \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1051 before the glossaries. If you want to use any entries \MessageBreak
1052 after any of the glossaries, you must use the standard \MessageBreak
1053 method of first defining the entry and then using the \MessageBreak
1054 entry with commands like \string\gls}%
1055 \@glstr@disabledflycommand
1056 }%
1057 \newcommand*{\@glstr@disabledflycommand}[2][]{##2}

```

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

```
1058 \let\GlsXtrEnableOnTheFly\relax
1059 }
1060 \@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.

```
1061 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}
```

Modify `\setglossarystyle` to set the above.

`etglossarystyle`

```
1062 \renewcommand*{\setglossarystyle}[1]{%
1063   \ifcsundef{@glsstyle@#1}%
1064   {%
1065     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
1066   }%
1067   {%
1068     \csname @glsstyle@#1\endcsname
```

Only set the current style if it exists.

```
1069   \protected@edef\@glxtr@current@style{#1}%
1070   }%
1071   \ifx\@glossary@default@style\relax
1072     \protected@edef\@glossary@default@style{#1}%
1073   \fi
1074 }
```

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

```
1075 \ifdef\@glossary@default@style
1076 {}
1077 {%
1078   \let\@glossary@default@style\relax
1079 }
```

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

```
1080 \ifdef\glslistdottedwidth
1081 {%
1082   \ifdim\glslistdottedwidth=.5\hsize
1083     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1084     \AtBeginDocument{%
1085       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1086         \setlength{\glslistdottedwidth}{.5\columnwidth}%
1087     \fi
```

```

1088 }%
1089 \fi
1090 }
1091 {}%

```

Similarly for `\glsdescwidth`:

`\glsdescwidth`

```

1092 \ifdef\glsdescwidth
1093 {%
1094   \ifdim\glsdescwidth=.6\hsize
1095     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1096     \AtBeginDocument{%
1097       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1098         \setlength{\glsdescwidth}{.6\columnwidth}%
1099       \fi
1100     }%
1101   \fi
1102 }
1103 {}%

```

and for `\glspagelistwidth`:

`lspagelistwidth`

```

1104 \ifdef\glspagelistwidth
1105 {%
1106   \ifdim\glspagelistwidth=.1\hsize
1107     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
1108     \AtBeginDocument{%
1109       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1110         \setlength{\glspagelistwidth}{.1\columnwidth}%
1111       \fi
1112     }%
1113   \fi
1114 }
1115 {}%

```

`aryentrynumbers` Has the nonnumberlist option been used?

```

1116 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1117 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1118   \glsnonnumberlistfalse
1119   \renewcommand*{\glossaryentrynumbers}[1]{%
1120     \ifglstryexists{\glscurrententrylabel}%
1121     {%
1122       \@glsxtrpreloctag
1123       \GlsXtrFormatLocationList{#1}%
1124       \@glsxtrpostloctag
1125       \gls@save@numberlist{#1}%
1126     }{}%
1127   }%

```

```

1128 \else
1129   \glsnonnumberlisttrue
1130   \renewcommand*{\glossaryentrynumbers}[1]{%
1131     \ifglsentryexists{\glscurrententrylabel}%
1132     {%
1133       \gls@save@numberlist{#1}%
1134     }{}%
1135   }%
1136 \fi

```

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

```

1137 \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`

```

1138 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
1139   \let\@glxtrpreloctag\@glxtrpreloctag
1140   \let\@glxtrpostloctag\@glxtrpostloctag
1141   \renewcommand*{\@glxtr@pagetag}{#1}%
1142   \renewcommand*{\@glxtr@pagetag}{#2}%
1143   \renewcommand*{\@glxtr@savepreloctag}[2]{%
1144     \csgdef{\@glxtr@preloctag##1}{##2}%
1145   }%
1146   \renewcommand*{\@glxtr@doloctag}{%
1147     \ifcsundef{\@glxtr@preloctag\glscurrententrylabel}%
1148     {%
1149       \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
1150         Rerun required}%
1151     }%
1152     {%
1153       \csuse{\@glxtr@preloctag\glscurrententrylabel}%
1154     }%
1155   }%
1156 }
1157 \@onlypreamble\GlsXtrEnablePreLocationTag

```

`\glxtrpreloctag`

```

1158 \newcommand*{\@glxtrpreloctag}{%
1159   \let\@glxtr@org@delimN\delimN
1160   \let\@glxtr@org@delimR\delimR
1161   \let\@glxtr@org@glsgignore\glsgignore

```

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

```

1162 \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1163 \renewcommand*{\delimN}{\%
1164 \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1165 \@glsxtr@org@delimN}%
1166 \renewcommand*{\delimR}{\%
1167 \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1168 \@glsxtr@org@delimR}%
1169 \renewcommand*{\glsignore}[1]{\%
1170 \gdef\@glsxtr@thisloctag{\relax}%
1171 \@glsxtr@org@glsignore{##1}}%
1172 \@glsxtr@doloctag
1173 }

```

glsxtrpreloctag

```
1174 \newcommand*{\@glsxtrpreloctag}{}
```

@glsxtr@pagetag

```
1175 \newcommand*{\@glsxtr@pagetag}{\%}
```

glsxtr@pagetag

```
1176 \newcommand*{\@glsxtr@pagetag}{\%}
```

lsxtrpostloctag

```

1177 \newcommand*{\@glsxtrpostloctag}{\%
1178 \let\delimN\@glsxtr@org@delimN
1179 \let\delimR\@glsxtr@org@delimR
1180 \let\glsignore\@glsxtr@org@glsignore
1181 \protected@write\@auxout{\%
1182 {\string\@glsxtr@savepreloctag{\glscurrententrylabel}\@glsxtr@thisloctag}}%
1183 }

```

lsxtrpostloctag

```
1184 \newcommand*{\@glsxtrpostloctag}{}
```

lsxtr@preloctag

```

1185 \newcommand*{\@glsxtr@savepreloctag}[2]{\%
1186 \protected@write\@auxout{\%
1187 \string\providecommand\string\@glsxtr@savepreloctag[2]{}}

```

glsxtr@doloctag

```
1188 \newcommand*{\@glsxtr@doloctag}{}
```

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

```

1189 \renewcommand*{\KV@printgloss@nonumberlist}[1]{\%
1190 \XKV@plfalse
1191 \XKV@sttrue
1192 \XKV@checkchoice[\XKV@resa]{#1}{true,false}%

```

```

1193 {%
1194   \csname glsnonumberlist\XKV@resa\endcsname
1195   \ifglsnonumberlist
1196     \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1197   \else
1198     \def\glossaryentrynumbers##1{%
1199       \@glstrpreloctag
1200       \GlsXtrFormatLocationList{##1}%
1201       \@glstrpostloctag
1202       \gls@save@numberlist{##1}}%
1203   \fi
1204 }%
1205 }

```

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.

```

1206 \renewcommand*{\glsentryfmt}{%
1207   \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}{}}{%
1208     \glsifregular{\glslabel}%
1209     {\glstrregularfont{\glsgenentryfmt}}%
1210   {%
1211     \ifglshasshort{\glslabel}%
1212     {\glstrgenabbrvfmt}%
1213     {\glstrregularfont{\glsgenentryfmt}}%
1214   }%
1215 }

```

`\glstrregularfont` Font used for regular entries.

```
1216 \newcommand*{\glstrregularfont}[1]{#1}
```

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

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

```
1217 \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.

```

1218 \@glxstr@record{#2}{#3}%
1219 \glsoifexists{#3}%
1220 {%

```

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).

```

1221 \let\glxstrorg@ifKV@glslink@hyper@ifKV@glslink@hyper
1222 \let\do@gls@link@checkfirsthyper@gls@link@nocheckfirsthyper
1223 \def\glscustomtext{#4}%
1224 \@glxstr@field@linkdefs
1225 #1%
1226 \@gls@link[#2]{#3}{#4}%
1227 \let@ifKV@glslink@hyper\glxstrorg@ifKV@glslink@hyper
1228 }%
1229 \glspostlinkhook
1230 }

```

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

\@gls@ Save the original definition and redefine.

```

1231 \let\@glxstr@org@gls@\@gls@
1232 \def\@gls@#1#2{%
1233 \@glxstr@record{#1}{#2}%
1234 \@glxstr@org@gls@{#1}{#2}%
1235 }%

```

\@glspl@ Save the original definition and redefine.

```

1236 \let\@glxstr@org@glspl@\@glspl@
1237 \def\@glspl@#1#2{%
1238 \@glxstr@record{#1}{#2}%
1239 \@glxstr@org@glspl@{#1}{#2}%
1240 }%

```

\@Gls@ Save the original definition and redefine.

```

1241 \let\@glxstr@org@Gls@\@Gls@
1242 \def\@Gls@#1#2{%
1243 \@glxstr@record{#1}{#2}%
1244 \@glxstr@org@Gls@{#1}{#2}%
1245 }%

```

\@Glspl@ Save the original definition and redefine.

```

1246 \let\@glxstr@org@Glspl@\@Glspl@
1247 \def\@Glspl@#1#2{%
1248 \@glxstr@record{#1}{#2}%
1249 \@glxstr@org@Glspl@{#1}{#2}%
1250 }%

```


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

```
1251 \let\@glxtr@org@GLS@\@GLS@
1252 \def\@GLS@#1#2{%
1253   \@glxtr@record{#1}{#2}%
1254   \@glxtr@org@GLS@{#1}{#2}%
1255 }%
```

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

```
1256 \let\@glxtr@org@GLSpl@\@GLSpl@
1257 \def\@GLSpl@#1#2{%
1258   \@glxtr@record{#1}{#2}%
1259   \@glxtr@org@GLSpl@{#1}{#2}%
1260 }%
```

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

```
1261 \renewcommand*\@glsdisp}[3][]{%
1262   \@glxtr@record{#1}{#2}%
1263   \glsdofexists{#2}{%
1264     \let\do@gl@link@checkfirsthyper\@gl@link@checkfirsthyper
1265     \let\gl@sifplural\@secondoftwo
1266     \let\gl@scapscase\@firstofthree
1267     \def\glscustomtext{#3}%
1268     \def\glinsert{}%
1269     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
1270     \@gl@link[#1]{#2}{\@glo@text}%
1271     \ifKV@gl@link@local
1272       \gl@localunset{#2}%
1273     \else
1274       \gl@unset{#2}%
1275     \fi
1276   }%
1277   \glspostlinkhook
1278 }
```

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

```
1279 \renewcommand*\@gl@link}[3][]{%
1280   \@glxtr@record{#1}{#2}%
1281   \glsdofexistsordo{#2}%
1282   {%
1283     \let\do@gl@link@checkfirsthyper\relax
1284     \@gl@link[#1]{#2}{#3}%
1285   }%
1286   {%
1287     \glstextformat{#3}%
1288   }%
1289   \glspostlinkhook
1290 }
```

`\glsadd` Redefine to include `\@glstr@record`

```
1291 \renewrobustcmd*{\glsadd}[2][\]{%
1292   \@gls@adjustmode
1293   \@glstr@record{#1}{#2}%
1294   \glsdoifexists{#2}%
1295   {%
1296     \def\@glsnumberformat{glsnumberformat}%
1297     \edef\@gls@counter{\csname glo@glsetoklabel{#2}@counter\endcsname}%
1298     \setkeys{glossadd}{#1}%
1299     \@gls@saveentrycounter
1300     \@do@wrglossary{#2}%
1301   }%
1302 }
```

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

```
1303 \newcommand*{\@glstr@field@linkdefs}{%
1304   \let\glstrifwasfirstuse\@secondoftwo
1305   \let\glssifplural\@secondoftwo
1306   \let\glscapscase\@firstofthree
1307   \let\glinsert\@empty
1308 }
```

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

`assignfieldfont`

```
1309 \newcommand*{\glstrassignfieldfont}[1]{%
1310   \ifglsentryexists{#1}%
1311   {%
1312     \ifglshasshort{#1}%
1313     {%
1314       \glsetabbrvfmt{\glscategory{#1}}%
1315       \glssifregular{#1}%
1316       {\let\@gls@field@font\glstrregularfont}%
1317       {\let\@gls@field@font\@firstofone}%
1318     }%
1319     {%
1320       \glssifnotregular{#1}%
1321       {\let\@gls@field@font\@firstofone}%
1322       {\let\@gls@field@font\glstrregularfont}%
1323     }%
1324   }%
1325   {%
1326     \let\@gls@field@font\@gobble
1327   }%
1328 }
```

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

```
1329 \def\@glstext@#1#2[#3]{%
```

```

1330 \glstrassignfieldfont{#2}%
1331 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
1332 }

```

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

```

1333 \def\@GLStext@#1#2[#3]{%
1334 \glstrassignfieldfont{#2}%
1335 \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
1336 {\@gls@field@font{\GLSaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
1337 }

```

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

```

1338 \def\@Glstext@#1#2[#3]{%
1339 \glstrassignfieldfont{#2}%
1340 \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
1341 {\@gls@field@font{\GLSaccesstext{#2}#3}}%
1342 }

```

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.

`checknohyperfirst`

```

1343 \newcommand*\glstrchecknohyperfirst[1]{%
1344 \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
1345 }

```

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

```

1346 \def\@glsfirst@#1#2[#3]{%
1347 \glstrassignfieldfont{#2}%
    Ensure that \glsfirst honours the nohyperfirst attribute.
1348 \@gls@field@link
1349 [\let\glstrifwasfirstuse\@firstoftwo
1350 \glstrchecknohyperfirst{#2}%
1351 ]{#1}{#2}%
1352 {\@gls@field@font{\glsaccessfirst{#2}#3}}%
1353 }

```

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

```

1354 \def\@Glsfirst@#1#2[#3]{%
1355 \glstrassignfieldfont{#2}%
    Ensure that \Glsfirst honours the nohyperfirst attribute.
1356 \@gls@field@link
1357 [\let\glstrifwasfirstuse\@firstoftwo
1358 \let\glscapscase\@secondofthree
1359 \glstrchecknohyperfirst{#2}%
1360 ]%
1361 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}#3}}%
1362 }

```

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

```
1363 \def\@GLSfirst@#1#2[#3]{%
1364   \glstrassignfieldfont{#2}%
      Ensure that \@GLSfirst honours the nohyperfirst attribute.
1365   \@gls@field@link
1366   [\let\glstrifwasfirstuse\@firstoftwo
1367   \let\glscapscase\@thirdofthree
1368   \glstrchecknohyperfirst{#2}%
1369   ]%
1370   {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1371 }
```

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

```
1372 \def\@glsplural@#1#2[#3]{%
1373   \glstrassignfieldfont{#2}%
1374   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1375   {\@gls@field@font{\glsaccessplural{#2}#3}}%
1376 }
```

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

```
1377 \def\@Glsplural@#1#2[#3]{%
1378   \glstrassignfieldfont{#2}%
1379   \@gls@field@link
1380   [\let\glsifplural\@firstoftwo
1381   \let\glscapscase\@secondofthree
1382   ]%
1383   {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1384 }
```

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

```
1385 \def\@GLSplural@#1#2[#3]{%
1386   \glstrassignfieldfont{#2}%
1387   \@gls@field@link
1388   [\let\glsifplural\@firstoftwo
1389   \let\glscapscase\@thirdofthree
1390   ]%
1391   {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1392 }
```

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

```
1393 \def\glsfirstplural@#1#2[#3]{%
1394   \glstrassignfieldfont{#2}%
      Ensure that \glsfirstplural honours the nohyperfirst attribute.
1395   \@gls@field@link
1396   [\let\glstrifwasfirstuse\@firstoftwo
1397   \let\glsifplural\@firstoftwo
1398   \glstrchecknohyperfirst{#2}%
```

```

1399 ]%
1400 {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1401 }

```

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

```

1402 \def\@Glsfirstplural@#1#2[#3]{%
1403   \glsxtrassignfieldfont{#2}%
      Ensure that \glsfirstplural honours the nohyperfirst attribute.
1404   \@gls@field@link
1405   [\let\glsxtrifwasfirstuse\@firstoftwo
1406    \let\glsifplural\@firstoftwo
1407    \let\glscapscase\@secondofthree
1408    \glsxtrchecknohyperfirst{#2}%
1409   ]%
1410   {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1411 }

```

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

```

1412 \def\@GLSfirstplural@#1#2[#3]{%
1413   \glsxtrassignfieldfont{#2}%
      Ensure that \glsfirstplural honours the nohyperfirst attribute.
1414   \@gls@field@link
1415   [\let\glsxtrifwasfirstuse\@firstoftwo
1416    \let\glsifplural\@firstoftwo
1417    \let\glscapscase\@thirdofthree
1418    \glsxtrchecknohyperfirst{#2}%
1419   ]%
1420   {#1}{#2}%
1421   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
1422 }

```

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

```

1423 \def\@glsname@#1#2[#3]{%
1424   \glsxtrassignfieldfont{#2}%
1425   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
1426 }

```

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

```

1427 \def\@Glsname@#1#2[#3]{%
1428   \glsxtrassignfieldfont{#2}%
1429   \@gls@field@link
1430   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1431   {\@gls@field@font{\Glsaccessname{#2}#3}}%
1432 }

```

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

```

1433 \def\@GLSname@#1#2[#3]{%

```

```

1434 \glstrassignfieldfont{#2}%
1435 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1436   {#1}{#2}%
1437   {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
1438 }

```

\@glsdesc@

```

1439 \def\@glsdesc@#1#2[#3]{%
1440   \glstrassignfieldfont{#2}%
1441   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}}%
1442 }

```

\@GLSdesc@ First letter uppercase version.

```

1443 \def\@GLSdesc@#1#2[#3]{%
1444   \glstrassignfieldfont{#2}%
1445   \@gls@field@link
1446   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1447   {\@gls@field@font{\GLSaccessdesc{#2}#3}}}%
1448 }

```

\@GLSdesc@ All uppercase version.

```

1449 \def\@GLSdesc@#1#2[#3]{%
1450   \glstrassignfieldfont{#2}%
1451   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1452   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
1453 }

```

@glsdescplural@ No case-changing version.

```

1454 \def\@glsdescplural@#1#2[#3]{%
1455   \glstrassignfieldfont{#2}%
1456   \@gls@field@link
1457   [\let\glscapscase\@secondoftwo
1458   \let\glsifplural\@firstoftwo
1459   ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}}%
1460 }

```

@Glsdescplural@ First letter uppercase version.

```

1461 \def\@Glsdescplural@#1#2[#3]{%
1462   \glstrassignfieldfont{#2}%
1463   \@gls@field@link
1464   [\let\glscapscase\@secondoftwo
1465   \let\glsifplural\@firstoftwo
1466   ]{#1}{#2}{\@gls@field@font{\GLSaccessdescplural{#2}#3}}}%
1467 }

```

@GLSdescplural@ All uppercase version.

```

1468 \def\@GLSdesc@#1#2[#3]{%
1469   \glstrassignfieldfont{#2}%

```

```

1470 \@gls@field@link
1471 [\let\glscapscase\@thirdoftwo
1472 \let\glsifplural\@firstoftwo
1473 ]%
1474   {#1}{#2}%
1475   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
1476 }

```

\@glssymbol@

```

1477 \def\@glssymbol@#1#2[#3]{%
1478   \glstrassignfieldfont{#2}%
1479   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
1480 }

```

\@Glssymbol@ First letter uppercase version.

```

1481 \def\@Glssymbol@#1#2[#3]{%
1482   \glstrassignfieldfont{#2}%
1483   \@gls@field@link
1484   [\let\glscapscase\@secondoftwo]%
1485   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
1486 }

```

\@GLSsymbol@ All uppercase version.

```

1487 \def\@GLSsymbol@#1#2[#3]{%
1488   \glstrassignfieldfont{#2}%
1489   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1490   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
1491 }

```

lssymbolplural@ No case-changing version.

```

1492 \def\@lssymbolplural@#1#2[#3]{%
1493   \glstrassignfieldfont{#2}%
1494   \@gls@field@link
1495   [\let\glscapscase\@secondoftwo
1496   \let\glsifplural\@firstoftwo
1497   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
1498 }

```

lssymbolplural@ First letter uppercase version.

```

1499 \def\@Glssymbolplural@#1#2[#3]{%
1500   \glstrassignfieldfont{#2}%
1501   \@gls@field@link
1502   [\let\glscapscase\@secondoftwo
1503   \let\glsifplural\@firstoftwo
1504   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
1505 }

```

LSsymbolplural@ All uppercase version.

```

1506 \def\@GLSSymbol@#1#2[#3]{%
1507   \glstrassignfieldfont{#2}%
1508   \@gls@field@link
1509   [\let\glscapscase\@thirdoftwo
1510    \let\glsifplural\@firstoftwo
1511   ]%
1512   {#1}{#2}%
1513   {\@gls@field@font{\@GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
1514 }

```

\@Glsuseri@ First letter uppercase version.

```

1515 \def\@Glsuseri@#1#2[#3]{%
1516   \glstrassignfieldfont{#2}%
1517   \@gls@field@link
1518   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1519   {\@gls@field@font{\@Glsentryuseri{#2}#3}}%
1520 }

```

\@GLSuseri@ All uppercase version.

```

1521 \def\@GLSuseri@#1#2[#3]{%
1522   \glstrassignfieldfont{#2}%
1523   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1524   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\@Glsentryuseri{#2}#3}}}%
1525 }

```

\@Glsuserii@ First letter uppercase version.

```

1526 \def\@Glsuserii@#1#2[#3]{%
1527   \glstrassignfieldfont{#2}%
1528   \@gls@field@link
1529   [\let\glscapscase\@secondoftwo]%
1530   {#1}{#2}{\@gls@field@font{\@Glsentryuserii{#2}#3}}%
1531 }

```

\@GLSuserii@ All uppercase version.

```

1532 \def\@GLSuserii@#1#2[#3]{%
1533   \glstrassignfieldfont{#2}%
1534   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1535   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\@Glsentryuserii{#2}#3}}}%
1536 }

```

\@Glsuseriii@ First letter uppercase version.

```

1537 \def\@Glsuseriii@#1#2[#3]{%
1538   \glstrassignfieldfont{#2}%
1539   \@gls@field@link
1540   [\let\glscapscase\@secondoftwo]%
1541   {#1}{#2}{\@gls@field@font{\@Glsentryuseriii{#2}#3}}%
1542 }

```


\@GLSuseriii@ All uppercase version.

```
1543 \def\@GLSuseriii@#1#2[#3]{%
1544   \glstrassignfieldfont{#2}%
1545   \@gls@field@link[\let\glscapscale\@thirdoftwo]%
1546   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
1547 }
```

\@Glsuseriv@ First letter uppercase version.

```
1548 \def\@Glsuseriv@#1#2[#3]{%
1549   \glstrassignfieldfont{#2}%
1550   \@gls@field@link
1551   [\let\glscapscale\@secondoftwo]%
1552   {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}}%
1553 }
```

\@GLSuseriv@ All uppercase version.

```
1554 \def\@GLSuseriv@#1#2[#3]{%
1555   \glstrassignfieldfont{#2}%
1556   \@gls@field@link[\let\glscapscale\@thirdoftwo]%
1557   {#1}{#2}%
1558   {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
1559 }
```

\@Glsuserv@ First letter uppercase version.

```
1560 \def\@Glsuserv@#1#2[#3]{%
1561   \glstrassignfieldfont{#2}%
1562   \@gls@field@link
1563   [\let\glscapscale\@secondoftwo]%
1564   {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}}%
1565 }
```

\@GLSuserv@ All uppercase version.

```
1566 \def\@GLSuserv@#1#2[#3]{%
1567   \glstrassignfieldfont{#2}%
1568   \@gls@field@link[\let\glscapscale\@thirdoftwo]%
1569   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
1570 }
```

\@Glsuservi@ First letter uppercase version.

```
1571 \def\@Glsuservi@#1#2[#3]{%
1572   \glstrassignfieldfont{#2}%
1573   \@gls@field@link
1574   [\let\glscapscale\@secondoftwo]%
1575   {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}}%
1576 }
```

\@GLSuservi@ All uppercase version.

```
1577 \def\@GLSuservi@#1#2[#3]{%
```

```

1578 \glstrassignfieldfont{#2}%
1579 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1580 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glstryuservi{#2}{#3}}}%
1581 }

```

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

\@acrshort No case change.

```

1582 \def\@acrshort#1#2[#3]{%
1583 \glsoifexists{#2}%
1584 {%
1585 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1586 \let\glstrifwasfirstuse\@secondoftwo
1587 \let\glsifplural\@secondoftwo
1588 \let\glscapscase\@firstofthree
1589 \let\glsinsert\@empty
1590 \def\glscustomtext{%
1591 \acronymfont{\glsaccessshort{#2}}#3%
1592 }%
1593 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1594 }%
1595 \glspostlinkhook
1596 }

```

\@Acrshort First letter uppercase.

```

1597 \def\@Acrshort#1#2[#3]{%
1598 \glsoifexists{#2}%
1599 {%
1600 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1601 \let\glstrifwasfirstuse\@secondoftwo
1602 \let\glsifplural\@secondoftwo
1603 \let\glscapscase\@secondofthree
1604 \let\glsinsert\@empty
1605 \def\glscustomtext{%
1606 \acronymfont{\Glsaccessshort{#2}}#3%
1607 }%
1608 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1609 }%
1610 \glspostlinkhook
1611 }

```

\@ACRshort All uppercase.

```

1612 \def\@ACRshort#1#2[#3]{%
1613 \glsoifexists{#2}%
1614 {%
1615 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1616 \let\glstrifwasfirstuse\@secondoftwo
1617 \let\glsifplural\@secondoftwo

```

```

1618 \let\glscapscase\@thirdofthree
1619 \let\glsinsert\@empty
1620 \def\glscustomtext{%
1621     \mfirstucMakeUppercase{\acronymfont{\glsaccessshort{#2}}#3}%
1622 }%
1623 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1624 }%
1625 \glspostlinkhook
1626 }

```

\@acrshortpl No case change.

```

1627 \def\@acrshortpl#1#2[#3]{%
1628     \glsdoifexists{#2}%
1629     {%
1630         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1631         \let\glxtrifwasfirstuse\@secondoftwo
1632         \let\glsifplural\@firstoftwo
1633         \let\glscapscase\@firstofthree
1634         \let\glsinsert\@empty
1635         \def\glscustomtext{%
1636             \acronymfont{\glsaccessshortpl{#2}}#3%
1637         }%
1638         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1639     }%
1640     \glspostlinkhook
1641 }

```

\@Acrshortpl First letter uppercase.

```

1642 \def\@Acrshortpl#1#2[#3]{%
1643     \glsdoifexists{#2}%
1644     {%
1645         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1646         \let\glxtrifwasfirstuse\@secondoftwo
1647         \let\glsifplural\@firstoftwo
1648         \let\glscapscase\@secondofthree
1649         \let\glsinsert\@empty
1650         \def\glscustomtext{%
1651             \acronymfont{\Glsaccessshortpl{#2}}#3%
1652         }%
1653         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1654     }%
1655     \glspostlinkhook
1656 }

```

\@ACRshortpl All uppercase.

```

1657 \def\@ACRshortpl#1#2[#3]{%
1658     \glsdoifexists{#2}%
1659     {%
1660         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```

```

1661 \let\glxtrifwasfirstuse\@secondoftwo
1662 \let\glsifplural\@firstoftwo
1663 \let\glscapscase\@thirdofthree
1664 \let\glsinsert\@empty
1665 \def\glscustomtext{%
1666     \mfirstucMakeUppercase{\acronymfont{\glsaccessshortpl{#2}}#3}%
1667 }%
1668 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1669 }%
1670 \glspostlinkhook
1671 }

```

\@acrlong No case change.

```

1672 \def\@acrlong#1#2[#3]{%
1673     \glsdoifexists{#2}%
1674     {%
1675         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1676         \let\glxtrifwasfirstuse\@secondoftwo
1677         \let\glsifplural\@secondoftwo
1678         \let\glscapscase\@firstofthree
1679         \let\glsinsert\@empty
1680         \def\glscustomtext{%
1681             \acronymfont{\glsaccesslong{#2}}#3%
1682         }%
1683         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1684     }%
1685     \glspostlinkhook
1686 }

```

\@Acrlong First letter uppercase.

```

1687 \def\@Acrlong#1#2[#3]{%
1688     \glsdoifexists{#2}%
1689     {%
1690         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1691         \let\glxtrifwasfirstuse\@secondoftwo
1692         \let\glsifplural\@secondoftwo
1693         \let\glscapscase\@secondofthree
1694         \let\glsinsert\@empty
1695         \def\glscustomtext{%
1696             \acronymfont{\Glsaccesslong{#2}}#3%
1697         }%
1698         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1699     }%
1700     \glspostlinkhook
1701 }

```

\@ACRlong All uppercase.

```

1702 \def\@ACRlong#1#2[#3]{%
1703     \glsdoifexists{#2}%

```

```

1704 {%
1705   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1706   \let\glxtrifwasfirstuse\@secondoftwo
1707   \let\gl@ifplural\@secondoftwo
1708   \let\gl@scapscase\@thirdofthree
1709   \let\gl@insert\@empty
1710   \def\glscustomtext{%
1711     \mfirstucMakeUppercase{\acronymfont{\gl@accesslong{#2}}#3}%
1712   }%
1713   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1714 }%
1715 \glspostlinkhook
1716 }

```

\@acrlongpl No case change.

```

1717 \def\@acrlongpl#1#2[#3]{%
1718   \gl@ifexists{#2}%
1719   {%
1720     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1721     \let\glxtrifwasfirstuse\@secondoftwo
1722     \let\gl@ifplural\@firstoftwo
1723     \let\gl@scapscase\@firstofthree
1724     \let\gl@insert\@empty
1725     \def\glscustomtext{%
1726       \acronymfont{\gl@accesslongpl{#2}}#3%
1727     }%
1728     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1729   }%
1730   \glspostlinkhook
1731 }

```

\@Acrlongpl First letter uppercase.

```

1732 \def\@Acrlongpl#1#2[#3]{%
1733   \gl@ifexists{#2}%
1734   {%
1735     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1736     \let\glxtrifwasfirstuse\@secondoftwo
1737     \let\gl@ifplural\@firstoftwo
1738     \let\gl@scapscase\@secondofthree
1739     \let\gl@insert\@empty
1740     \def\glscustomtext{%
1741       \acronymfont{\Gls@accesslongpl{#2}}#3%
1742     }%
1743     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1744   }%
1745   \glspostlinkhook
1746 }

```

\@ACRlongpl All uppercase.

```

1747 \def\ACRlongpl#1#2[#3]{%
1748   \glsdoifexists{#2}%
1749   {%
1750     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1751     \let\glxtrifwasfirstuse\@secondoftwo
1752     \let\gl@ifplural\@firstoftwo
1753     \let\gl@scapscase\@thirdofthree
1754     \let\gl@insert\@empty
1755     \def\glscustomtext{%
1756       \mfirstucMakeUppercase{\acronymfont{\gl@accesslongpl{#2}}#3}%
1757     }%
1758     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1759   }%
1760   \glspostlinkhook
1761 }

```

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

\@gl@saddkey

```

1762 \renewcommand*{\@gl@saddkey}[7]{%
1763   \key@ifundefined{glossentry}{#1}%
1764   {%
1765     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1766     \appto\@gl@keymap{, {#1}{#1}}%
1767     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
1768     \appto\@newglossaryentryposthook{%
1769       \letcs{\@glo@tmp}{@glo@#1}%
1770       \gl@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1771     }%
1772     \newcommand*{#3}[1]{\@gl@entry@field{##1}{#1}}%
1773     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

1774   \ifcsdef{@gl@user@#1@}%
1775   {%
1776     \PackageError{glossaries}%
1777     {Can't define '\string#5' as helper command
1778     '\expandafter\string\csname @gl@user@#1@\endcsname' already
1779     exists}%
1780   }%
1781 }%
1782 {%
1783   \expandafter\newcommand\expandafter*\expandafter
1784   {\csname @gl@user@#1\endcsname}[2][ ]{%
1785     \new@ifnextchar[%
1786       {\csuse{@gl@user@#1@}{##1}{##2}}%
1787       {\csuse{@gl@user@#1@}{##1}{##2} [ ]}}%
1788   \csdef{@gl@user@#1@}##1##2[##3]{%
1789     \@gl@field@link{##1}{##2}{#3{##2}##3}%

```

```

1790 }%
1791 \newrobustcmd*{#5}{%
1792   \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
1793 }%

```

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

```

1794 \ifcsdef{@Gls@user@#1@}%
1795 {%
1796   \PackageError{glossaries}%
1797   {Can't define '\string#6' as helper command
1798     '\expandafter\string\csname @Gls@user@#1\endcsname' already
1799     exists}%
1800 }%
1801 }%
1802 {%
1803   \expandafter\newcommand\expandafter*\expandafter
1804     {\csname @Gls@user@#1\endcsname}[2][{}]{%
1805     \new@ifnextchar[%
1806       {\csuse{@Gls@user@#1@}{##1}{##2}}%
1807       {\csuse{@Gls@user@#1@}{##1}{##2}[]}}%
1808   \csdef{@Gls@user@#1@}##1##2[##3]{%
1809     \@gls@field@link[\let\glscaps@case\@secondofthree]%
1810     {##1}{##2}{#4{##2}##3}%
1811   }%
1812   \newrobustcmd*{#6}{%
1813     \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
1814 }%

```

Finally the all caps version (modified):

```

1815 \ifcsdef{@GLS@user@#1@}%
1816 {%
1817   \PackageError{glossaries}%
1818   {Can't define '\string#7' as helper command
1819     '\expandafter\string\csname @GLS@user@#1\endcsname' already
1820     exists}%
1821 }%
1822 }%
1823 {%
1824   \expandafter\newcommand\expandafter*\expandafter
1825     {\csname @GLS@user@#1\endcsname}[2][{}]{%
1826     \new@ifnextchar[%
1827       {\csuse{@GLS@user@#1@}{##1}{##2}}%
1828       {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%
1829   \csdef{@GLS@user@#1@}##1##2[##3]{%
1830     \@gls@field@link[\let\glscaps@case\@thirdofthree]%
1831     {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1832   }%
1833   \newrobustcmd*{#7}{%
1834     \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
1835 }%

```

```

1836 }%
1837 {%
1838   \PackageError{glossaries-extra}{Key ‘#1’ already exists}{}%
1839 }%
1840 }

```

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

```

1841 \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.

```

1842 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
1843 \renewcommand*{\@gls@link@checkfirsthyper}{%

```

\ifglsused isn't useful in the post link hook as it's already been unset by then, so define a command that can be used in the post link hook. Since \@gls@link@checkfirsthyper is only used by commands like \gls but not by other commands, this seems the best place to put it.

```

1844   \ifglsused{\glslabel}%
1845   {\let\glsxtrifwasfirstuse\@secondoftwo}
1846   {\let\glsxtrifwasfirstuse\@firstoftwo}%

```

Store the category label for convenience.

```

1847   \edef\glscategorylabel{\glscategory{\glslabel}}%
1848   \ifglsused{\glslabel}%
1849   {%
1850     \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
1851     {\KV@glslink@hyperfalse}{}%
1852   }%
1853   {%
1854     \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
1855     {\KV@glslink@hyperfalse}{}%
1856   }%
1857   \glslinkcheckfirsthyperhook
1858 }

```

ablehyperinlist This command was introduced in glossaries v4.19. If it hasn't been defined, we're using an earlier version, in which case the nohyper attribute can't be implemented.

```

1859 \ifdef\do@glsdisablehyperinlist
1860 {%
1861   \let\@glsxtr@do@glsdisablehyperinlist\do@glsdisablehyperinlist
1862   \renewcommand*{\do@glsdisablehyperinlist}{%
1863     \@glsxtr@do@glsdisablehyperinlist
1864     \glsifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}%
1865   }
1866 }
1867 {}

```

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


```

1868 \define@boolkey{glslink}{noindex}[true]{}
1869 \KV@glslink@noindexfalse

```

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

`lt@glslink@opts`

```

1870 \ifdef\@gls@setdefault@glslink@opts
1871 {
1872   \renewcommand*{\@gls@setdefault@glslink@opts}{%
1873     \KV@glslink@noindexfalse
1874     \@glsxtrsetaliasnoindex
1875   }
1876 }
1877 {

```

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

```

1878   \newcommand*{\@gls@setdefault@glslink@opts}{%
1879     \KV@glslink@noindexfalse
1880     \@glsxtrsetaliasnoindex
1881   }
1882   \preto\do@glsgldisablehyperinlist{\@gls@setdefault@glslink@opts}
1883 }

```

`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.)

```

1884 \providecommand*{\glsxtrsetaliasnoindex}{%
1885   \KV@glslink@noindextrue
1886 }

```

`setaliasnoindex`

```

1887 \newcommand*{\@glsxtrsetaliasnoindex}{%
1888   \ifglshasfield{alias}{\glslabel}%
1889   {%
1890     \let\glsxtrindexaliased\@glsxtrindexaliased
1891     \glsxtrsetaliasnoindex
1892     \let\glsxtrindexaliased\@no@glsxtrindexaliased
1893   }%
1894   {%
1895 }

```

`xtrindexaliased`

```

1896 \newcommand{\@glsxtrindexaliased}{%
1897   \ifKV@glslink@noindex
1898   \else
1899     \begingroup
1900     \def\@glsnumberformat{glsnumberformat}%
1901     \edef\@gls@counter{\csname glo@glsdetoklabel{\glslabel}@counter\endcsname}%
1902     \glsxtr@saveentrycounter

```

```

1903 \do@wrglossary{\glxtralias{\glslabel}}%
1904 \endgroup
1905 \fi
1906 }

```

xtrindexaliased

```

1907 \newcommand{\@no@glxtrindexaliased}{%
1908 \PackageError{glossaries-extra}{\string\glxtrindexaliased\space
1909 not permitted outside definition of \string\glxtrsetaliasnoindex}%
1910 {}%
1911 }

```

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

```

1912 \let\glxtrindexaliased\@no@glxtrindexaliased

```

tDefaultGlsOpts Set the default options for \glslink etc.

```

1913 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
1914 \renewcommand*{\@glsetdefault@glslink@opts}{%
1915 \setkeys{glslink}{#1}%
1916 \@glxtrsetaliasnoindex
1917 }%
1918 }

```

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

```

1919 \newcommand*{\glxtrifindexing}[2]{%
1920 \ifKV@glslink@noindex #2\else #1\fi
1921 }

```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```

1922 \renewcommand*{\glswriteentry}[2]{%
1923 \glxtrifindexing
1924 {%
1925 \ifglindexonlyfirst
1926 \ifglused{#1}
1927 {\glxtrdoautoindexname{#1}{dualindex}}%
1928 {#2}%
1929 \else
1930 \gl@ifattribute{#1}{indexonlyfirst}{true}%
1931 {\ifglused{#1}
1932 {\glxtrdoautoindexname{#1}{dualindex}}%
1933 {#2}}%
1934 {#2}%
1935 \fi
1936 }%
1937 {}%
1938 }

```

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

```

1939 \appto\@do@@wrglossary{\@glstr@do@@wrindex
1940   \glstrdowrglossaryhook{\@gls@label}}%
1941 }

```

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

Similarly for the “noidx” version:

s@noidxglossary

```

1942 \appto\gls@noidxglossary{\@glstr@do@@wrindex
1943   \glstrdowrglossaryhook{\@gls@label}}%
1944 }

```

xtr@do@@wrindex

```

1945 \newcommand*{\@glstr@do@@wrindex}{%
1946   \glstrdoautoindexname{\@gls@label}{dualindex}}%
1947 }

```

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

```

1948 \newcommand*{\glstrdowrglossaryhook}[1]{%

```

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

```

1949 \newcommand*{\@gls@alt@hyp@opt}[1]{%
1950   \let\glslinkvar\@firstofthree
1951   \let\@gls@hyp@opt@cs#1\relax
1952   \@ifstar{\s@gls@hyp@opt}%
1953   {\@ifnextchar+%
1954     {\@firstoftwo{\p@gls@hyp@opt}}%
1955     {%
1956       \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
1957       {\@firstoftwo{\@alt@gls@hyp@opt}}%
1958       {#1}%
1959     }%
1960   }%
1961 }

```

alt@gls@hyp@opt User version

```

1962 \newcommand*{\@alt@gls@hyp@opt}[1][ ]{%
1963   \let\glslinkvar\@firstofthree
1964   \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}

```

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

```

1965 \newcommand*{\@gls@alt@hyp@opt@char}{}

```

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

```

1966 \newcommand*{\@gls@alt@hyp@opt@keys}{}

```

rSetAltModifier

```
1967 \newcommand*{\GlsXtrSetAltModifier}[2]{%
1968   \let\@gls@hyp@opt\@gls@alt@hyp@opt
1969   \def\@gls@alt@hyp@opt@char{#1}%
1970   \def\@gls@alt@hyp@opt@keys{#2}%
1971 }
```

\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.

```
1972 \renewcommand*{\glsdohyperlink}[2]{%
1973   \glsattribute{\glslabel}{targeturl}%
1974   {%
1975     \glsattribute{\glslabel}{targetname}%
1976     {%
1977       \glsattribute{\glslabel}{targetcategory}%
1978       {%
1979         \hyperref{\glsattribute{\glslabel}{targeturl}}%
1980           {\glsattribute{\glslabel}{targetcategory}}%
1981           {\glsattribute{\glslabel}{targetname}}%
1982           {\glsxtrprotectlinks#2}}%
1983       }%
1984     {%
1985       \hyperref{\glsattribute{\glslabel}{targeturl}}%
1986         {}%
1987         {\glsattribute{\glslabel}{targetname}}%
1988         {\glsxtrprotectlinks#2}}%
1989     }%
1990   }%
1991   {%
1992     \href{\glsattribute{\glslabel}{targeturl}}%
1993       {\glsxtrprotectlinks#2}}%
1994   }%
1995 }%
1996 {%
```

Check for alias.

```
1997   \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
1998   \ifvoid\gloaliaslabel
1999   {%
2000     \hyperlink{#1}{\glsxtrprotectlinks#2}}%
2001   }%
2002   {%
```

Redirect link to the alias target.

```

2003     \hyperlink
2004     {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}}%
2005     {\glxtrprotectlinks#2}}}%
2006   }%
2007 }%
2008 }

```

`glsdisablehyper` Redefine in case we have an old version of glossaries.

```

2009 \ifundef\glsdonohyperlink
2010 {%
2011   \renewcommand{\glsdisablehyper}{%
2012     \KV@glslink@hyperfalse
2013     \let\@glslink\glsdonohyperlink
2014     \let\@glstarget\@secondoftwo
2015   }
2016 }
2017 {}

```

`glsdonohyperlink` This command was only introduced in glossaries v4.20, so it may not be defined. 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.

```

2018 \def\glsdonohyperlink#1#2{{\glxtrprotectlinks #2}}

```

Reset `\@glslink` with patched versions:

```

2019 \ifcsundef{hyperlink}%
2020 {%
2021   \let\@glslink\glsdonohyperlink
2022 }%
2023 {%
2024   \let\@glslink\glsdohyperlink
2025 }

```

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

```

2026 \newcommand*{\glxtrprotectlinks}{%
2027   \KV@glslink@hyperfalse
2028   \KV@glslink@noindextrue
2029   \let\@gls@\@glxtr@p@text@
2030   \let\@Gls@\@Glsxtr@p@text@
2031   \let\@GLS@\@GLSxtr@p@text@
2032   \let\@glspl@\@glxtr@p@plural@
2033   \let\@Glspl@\@Glsxtr@p@plural@
2034   \let\@GLSpl@\@GLSxtr@p@plural@
2035   \let\@glxtrshort@\@glxtr@p@short@
2036   \let\@Glsxtrshort@\@Glsxtr@p@short@
2037   \let\@GLSxtrshort@\@GLSxtr@p@short@
2038   \let\@glxtrlong@\@glxtr@p@long@
2039   \let\@Glsxtrlong@\@Glsxtr@p@long@

```

```

2040 \let\@GLSxtrlong\@GLSxtr@p@long@
2041 \let\@glxtrshortpl\@glxtr@p@shortpl@
2042 \let\@Glsxtrshortpl\@Glsxtr@p@shortpl@
2043 \let\@GLSxtrshortpl\@GLSxtr@p@shortpl@
2044 \let\@glxtrlongpl\@glxtr@p@longpl@
2045 \let\@Glsxtrlongpl\@Glsxtr@p@longpl@
2046 \let\@GLSxtrlongpl\@GLSxtr@p@longpl@
2047 \let\@acrshort\@glxtr@p@acrshort@
2048 \let\@Acrshort\@Glsxtr@p@acrshort@
2049 \let\@ACRshort\@GLSxtr@p@acrshort@
2050 \let\@acrshortpl\@glxtr@p@acrshortpl@
2051 \let\@Acrshortpl\@Glsxtr@p@acrshortpl@
2052 \let\@ACRshortpl\@GLSxtr@p@acrshortpl@
2053 \let\@acrlong\@glxtr@p@acrlong@
2054 \let\@Acrlong\@Glsxtr@p@acrlong@
2055 \let\@ACRlong\@GLSxtr@p@acrlong@
2056 \let\@acrlongpl\@glxtr@p@acrlongpl@
2057 \let\@Acrlongpl\@Glsxtr@p@acrlongpl@
2058 \let\@ACRlongpl\@GLSxtr@p@acrlongpl@
2059 }

```

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

@glxtr@p@text@

```
2060 \def\@glxtr@p@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}
```

@Glsxtr@p@text@

```
2061 \def\@Glsxtr@p@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtr@p@text@

```
2062 \def\@GLSxtr@p@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtr@p@plural@

```
2063 \def\@glxtr@p@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

@Glsxtr@p@plural@

```
2064 \def\@Glsxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

@GLSxtr@p@plural@

```
2065 \def\@GLSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glxtr@p@short@

```

2066 \def\@glxtr@p@short@#1#2[#3]{%
2067   {%
2068     \glssetabbrvfmt{\glscategory{#2}}%
2069     \glsabbrvfont{\glsentryshort{#2}}#3%
2070   }%
2071 }

```

Glsxtr@p@short@

```
2072 \def\@Glsxtr@p@short@#1#2[#3]{%
2073 {%
2074   \glsetabbrvfmt{\glscategory{#2}}%
2075   \glsabbrvfont{\Glsentryshort{#2}}#3%
2076 }%
2077 }
```

GLSxtr@p@short@

```
2078 \def\@GLSxtr@p@short@#1#2[#3]{%
2079 {%
2080   \glsetabbrvfmt{\glscategory{#2}}%
2081   \mfirstucMakeUppercase{\glsabbrvfont{\Glsentryshort{#2}}#3}%
2082 }%
2083 }
```

sxtr@p@shortpl@

```
2084 \def\@glxtr@p@shortpl@#1#2[#3]{%
2085 {%
2086   \glsetabbrvfmt{\glscategory{#2}}%
2087   \glsabbrvfont{\Glsentryshortpl{#2}}#3%
2088 }%
2089 }
```

sxtr@p@shortpl@

```
2090 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2091 {%
2092   \glsetabbrvfmt{\glscategory{#2}}%
2093   \glsabbrvfont{\Glsentryshortpl{#2}}#3%
2094 }%
2095 }
```

Sxtr@p@shortpl@

```
2096 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2097 {%
2098   \glsetabbrvfmt{\glscategory{#2}}%
2099   \mfirstucMakeUppercase{\glsabbrvfont{\Glsentryshortpl{#2}}#3}%
2100 }%
2101 }
```

@glxtr@p@long@

```
2102 \def\@glxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@Glsxtr@p@long@

```
2103 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@GLSxtr@p@long@

```
2104 \def\@GLSxtr@p@long@#1#2[#3]{%
2105   {\mfirstucMakeUppercase{\glslongfont{\Glsentrylong{#2}}#3}}}
```

```

lsxtr@p@longpl@
2106 \def\@glxtr@p@longpl@#1#2[#3]{\glentrylongpl{#2}#3}}

lsxtr@p@longpl@
2107 \def\@Glsxtr@p@longpl@#1#2[#3]{\glslongfont{\glentrylongpl{#2}#3}}

LSxtr@p@longpl@
2108 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2109   {\mfirstucMakeUppercase{\glslongfont{\glentrylongpl{#2}#3}}}}

xtr@p@acrshort@
2110 \def\@glxtr@p@acrshort@#1#2[#3]{\acronymfont{\glentryshort{#2}#3}}

xtr@p@acrshort@
2111 \def\@Glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\glentryshort{#2}#3}}

xtr@p@acrshort@
2112 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2113   {\mfirstucMakeUppercase{\acronymfont{\glentryshort{#2}#3}}}}

r@p@acrshortpl@
2114 \def\@glxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\glentryshortpl{#2}#3}}

r@p@acrshortpl@
2115 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\glentryshortpl{#2}#3}}

r@p@acrshortpl@
2116 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2117   {\mfirstucMakeUppercase{\acronymfont{\glentryshortpl{#2}#3}}}}

sxtr@p@acrlong@
2118 \def\@glxtr@p@acrlong@#1#2[#3]{\glentrylong{#2}#3}}

sxtr@p@acrlong@
2119 \def\@Glsxtr@p@acrlong@#1#2[#3]{\glentrylong{#2}#3}}

Sxtr@p@acrlong@
2120 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2121   {\mfirstucMakeUppercase{\glentrylong{#2}#3}}}}

tr@p@acrlongpl@
2122 \def\@glxtr@p@acrlongpl@#1#2[#3]{\glentrylongpl{#2}#3}}

tr@p@acrlongpl@
2123 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\glentrylongpl{#2}#3}}

```


tr@p@acrlongpl@

```
2124 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2125 {\mfirstucMakeUppercase{\glstrylongpl{#2}#3}}}
```

Commands to minimise conflict.

\@glxtrp@opt

```
2126 \newcommand*{\@glxtrp@opt}{hyper=false,noindex}
```

\glxtrsetpopts Used in glossary to switch hyperlinks on for the \glxtrp type of commands.

```
2127 \newcommand*{\glxtrsetpopts}[1]{%
2128 \renewcommand*{\@glxtrp@opt}{#1}%
2129 }
```

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

```
2130 \newcommand*{\glossxtrsetpopts}{%
2131 \glxtrsetpopts{noindex}%
2132 }
```

\@@glxtrp

```
2133 \newrobustcmd*{\@@glxtrp}[2]{%
```

Add scope.

```
2134 {%
2135 \let\glspostlinkhook\relax
2136 \csname#1\expandafter\endcsname\expandafter[\@glxtrp@opt]{#2}[]%
2137 }%
2138 }
```

\@glxtrp

```
2139 \newrobustcmd*{\@glxtrp}[2]{%
2140 \ifcsdef{gls#1}%
2141 {%
2142 \@glxtrp{gls#1}{#2}%
2143 }%
2144 {%
2145 \ifcsdef{glsxtr#1}%
2146 {%
2147 \@glxtrp{glsxtr#1}{#2}%
2148 }%
2149 {%
2150 \PackageError{glossaries-extra}{‘#1’ not recognised by
2151 \string\glxtrp}{}%
2152 }%
2153 }%
2154 }
```

\@Glsxtrp

```
2155 \newrobustcmd*{\@Glsxtrp}[2]{%
```

```

2156 \ifcsdef{Gls#1}%
2157 {%
2158   \@@glxstrp{Gls#1}{#2}%
2159 }%
2160 {%
2161   \ifcsdef{Glsxtr#1}%
2162   {%
2163     \@@glxstrp{Glsxtr#1}{#2}%
2164   }%
2165   {%
2166     \PackageError{glossaries-extra}{‘#1’ not recognised by
2167       \string\Glsxtrp}{}%
2168   }%
2169 }%
2170 }

```

\@GLSxtrp

```

2171 \newrobustcmd*{\@GLSxtrp}[2]{%
2172   \ifcsdef{GLS#1}%
2173   {%
2174     \@@glxstrp{GLS#1}{#2}%
2175   }%
2176   {%
2177     \ifcsdef{GLSxtr#1}%
2178     {%
2179       \@@glxstrp{GLSxtr#1}{#2}%
2180     }%
2181     {%
2182       \PackageError{glossaries-extra}{‘#1’ not recognised by
2183         \string\GLSxtrp}{}%
2184     }%
2185   }%
2186 }

```

\glxstr@entry@p

```

2187 \newrobustcmd*{\glxstr@headentry@p}[2]{%
2188   \gl@ifattribute{#1}{headuc}{true}%
2189   {%
2190     \mfirstucMakeUppercase{\@gl@entry@field{#1}{#2}}%
2191   }%
2192   {%
2193     \@gl@entry@field{#1}{#2}%
2194   }%
2195 }

```

\glxstrp Not robust as it needs to expand somewhat.

```

2196 \ifdef\texorpdfstring
2197 {
2198   \newcommand{\glxstrp}[2]{%

```

```

2199 \protect\NoCaseChange
2200 {%
2201 \protect\texorpdfstring
2202 {%
2203 \protect\glstrifinmark
2204 {%
2205 \ifcsdef{glstrhead#1}%
2206 {%
2207 {\protect\csuse{glstrhead#1}{#2}}%
2208 }%
2209 {%
2210 \glstr@headentry@p{#2}{#1}%
2211 }%
2212 }%
2213 {%
2214 \@glstrp{#1}{#2}%
2215 }%
2216 }%
2217 {%
2218 \protect\@glstr@entry@field{#2}{#1}%
2219 }%
2220 }%
2221 }
2222 }
2223 {
2224 \newcommand{\glstrp}[2]{%
2225 \protect\NoCaseChange
2226 {%
2227 \protect\glstrifinmark
2228 {%
2229 \ifcsdef{glstrhead#1}%
2230 {%
2231 {\protect\csuse{glstrhead#1}}%
2232 }%
2233 {%
2234 \glstr@headentry@p{#2}{#1}%
2235 }%
2236 }%
2237 {%
2238 \@glstrp{#1}{#2}%
2239 }%
2240 }%
2241 }
2242 }

```

Provide short synonyms for the most common option.

`\glsp`

```

2243 \newcommand*{\glsp}{\glstrp{short}}

```

`\glspt`

```
2244 \newcommand*{\glspt}{\glsxtrp{text}}
```

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

```
2245 \ifdef\texorpdfstring
2246 {
2247   \newcommand{\Glsxtrp}[2]{%
2248     \protect\NoCaseChange
2249     {%
2250       \protect\texorpdfstring
2251       {%
2252         \protect\glsxtrifinmark
2253         {%
2254           \ifcsdef{Glsxtrhead#1}%
2255           {%
2256             {\protect\csuse{Glsxtrhead#1}{#2}}%
2257           }%
2258           {%
2259             \protect\@Gls@entry@field{#2}{#1}%
2260           }%
2261         }%
2262         {%
2263           \@Glsxtrp{#1}{#2}%
2264         }%
2265       }%
2266       {%
2267         \protect\@gls@entry@field{#2}{#1}%
2268       }%
2269     }%
2270   }
2271 }
2272 {
2273   \newcommand{\Glsxtrp}[2]{%
2274     \protect\NoCaseChange
2275     {%
2276       \protect\glsxtrifinmark
2277       {%
2278         \ifcsdef{Glsxtrhead#1}%
2279         {%
2280           {\protect\csuse{Glsxtrhead#1}}%
2281         }%
2282         {%
2283           \protect\@Gls@entry@field{#2}{#1}%
2284         }%
2285       }%
2286       {%
2287         \@Glsxtrp{#1}{#2}%
2288       }%
2289     }
2290 }
```

```

2289     }%
2290   }
2291 }

```

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

```

2292 \ifdef\teorpdfstring
2293 {
2294   \newcommand{\GLSxtrp}[2]{%
2295     \protect\NoCaseChange
2296     {%
2297       \protect\teorpdfstring
2298       {%
2299         \protect\glxtrifinmark
2300         {%
2301           \ifcsdef{GLSxtr#1}%
2302           {%
2303             {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2304           }%
2305           {%
2306             \protect\mfirstucMakeUppercase
2307             {%
2308               \protect\@gls@entry@field{#2}{#1}%
2309             }%
2310           }%
2311         }%
2312       }%
2313       \@GLSxtrp{#1}{#2}%
2314     }%
2315   }%
2316   {%
2317     \protect\@gls@entry@field{#2}{#1}%
2318   }%
2319 }%
2320 }
2321 }
2322 {
2323   \newcommand{\GLSxtrp}[2]{%
2324     \protect\NoCaseChange
2325     {%
2326       \protect\glxtrifinmark
2327       {%
2328         \ifcsdef{GLSxtr#1}%
2329         {%
2330           {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2331         }%
2332         {%
2333           \protect\mfirstucMakeUppercase
2334           {%
2335             \protect\@gls@entry@field{#2}{#1}%

```

```

2336         }%
2337     }%
2338 }%
2339 {%
2340     \@GLSxtrp{#1}{#2}%
2341 }%
2342 }%
2343 }
2344 }

```

1.3.5 Entry Counting

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

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

\@glsunset Global unset.

```

2345 \renewcommand*{\@glsunset}[1]{%
2346     \@glsunset{#1}%
2347     \glsxtrpostunset{#1}%
2348 }%

```

glsxtrpostunset

```

2349 \newcommand*{\glsxtrpostunset}[1]{}

```

\@glslocalunset Local unset.

```

2350 \renewcommand*{\@glslocalunset}[1]{%
2351     \@glslocalunset{#1}%
2352     \glsxtrpostlocalunset{#1}%
2353 }%

```

rpostlocalunset

```

2354 \newcommand*{\glsxtrpostlocalunset}[1]{}

```

\@glsreset Global reset.

```

2355 \renewcommand*{\@glsreset}[1]{%
2356     \@glsreset{#1}%
2357     \glsxtrpostreset{#1}%
2358 }%

```

glsxtrpostreset

```

2359 \newcommand*{\glsxtrpostreset}[1]{}

```

\@glslocalreset Local reset.

```

2360 \renewcommand*{\@glslocalreset}[1]{%
2361     \@glslocalreset{#1}%
2362     \glsxtrpostlocalreset{#1}%
2363 }%

```

rpostlocalreset

```
2364 \newcommand*{\glxtrpostlocalreset}[1]{}
```

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

```
2365 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

Enable entry counting:

```
2366 \glsenableentrycount
```

Redefine \gls etc:

```
2367 \renewcommand*{\gls}{\cglss}%
```

```
2368 \renewcommand*{\Gls}{\cGls}%
```

```
2369 \renewcommand*{\glspl}{\cglsspl}%
```

```
2370 \renewcommand*{\Glspl}{\cGlspl}%
```

```
2371 \renewcommand*{\GLS}{\cGLS}%
```

```
2372 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2373 \@glxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
2374 \let\GlsXtrEnableEntryCounting\@glxtr@setentrycountunsetattr
```

```
2375 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
```

```
2376 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
```

```
2377 can't be used with \string\GlsXtrEnableEntryCounting}%
```

```
2378 {Use one or other but not both commands}}%
```

```
2379 }
```

ycountunsetattr

```
2380 \newcommand*{\@glxtr@setentrycountunsetattr}[2]{%
```

```
2381 \@for\@glxtr@cat:=#1\do
```

```
2382 {%
```

```
2383 \ifdefempty{\@glxtr@cat}{}}%
```

```
2384 {%
```

```
2385 \glsssetcategoryattribute{\@glxtr@cat}{entrycount}{#2}%
```

```
2386 }%
```

```
2387 }%
```

```
2388 }
```

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

nableentrycount

```
2389 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
2390 \appto\@newglossaryentry@defcounters{\@@newglossaryentry@defcounters}%
```

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

```
2391 \renewcommand*{\gls@defdocnewglossaryentry}{%
```

```
2392 \renewcommand*{newglossaryentry}[2]{%
```

```

2393     \PackageError{glossaries}{\string\newglossaryentry\space
2394     may only be used in the preamble when entry counting has
2395     been activated}{If you use \string\glsenableentrycount\space
2396     you must place all entry definitions in the preamble not in
2397     the document environment}%
2398   }%
2399 }%

```

New commands to access new fields:

```

2400 \newcommand*{\glsentrycurrcount}[1]{%
2401   \ifcsundef{glo@\glsdetoklabel{##1}@currcount}%
2402   {0}{\@gls@entry@field{##1}{currcount}}}%
2403 }%
2404 \newcommand*{\glsentryprevcount}[1]{%
2405   \ifcsundef{glo@\glsdetoklabel{##1}@prevcount}%
2406   {0}{\@gls@entry@field{##1}{prevcount}}}%
2407 }%

```

Adjust post unset and reset:

```

2408 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
2409 \renewcommand*{\glsxtrpostunset}[1]{%
2410   \@glsxtr@entrycount@org@unset{##1}%
2411   \@gls@increment@currcount{##1}%
2412 }%
2413 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
2414 \renewcommand*{\glsxtrpostlocalunset}[1]{%
2415   \@glsxtr@entrycount@org@localunset{##1}%
2416   \@gls@local@increment@currcount{##1}%
2417 }%
2418 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
2419 \renewcommand*{\glsxtrpostreset}[1]{%
2420   \@glsxtr@entrycount@org@reset{##1}%
2421   \csgdef{glo@\glsdetoklabel{##1}@currcount}{0}%
2422 }%
2423 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
2424 \renewcommand*{\glsxtrpostlocalreset}[1]{%
2425   \@glsxtr@entrycount@org@localreset{##1}%
2426   \csdef{glo@\glsdetoklabel{##1}@currcount}{0}%
2427 }%

```

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

```

2428 \let\@cgl@\@cgl@
2429 \let\@cgl@pl@\@cgl@pl@
2430 \let\@cGLS@\@cGLS@
2431 \let\@cGL@pl@\@cGL@pl@
2432 \let\@cGLS@\@cGLS@
2433 \let\@cGL@pl@\@cGL@pl@

```

The rest is as the original definition.

```

2434 \AtEndDocument{\@gls@write@entrycounts}%

```



```

2435 \renewcommand*{\@gls@entry@count}[2]{%
2436   \csgdef{glo@glsetoklabel{##1}@prevcount}{##2}%
2437 }%
2438 \let\glsenableentrycount\relax
2439 \renewcommand*{\glsenableentryunitcount}{%
2440   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space
2441     can't be used with \string\glsenableentrycount}%
2442   {Use one or other but not both commands}%
2443 }%
2444 }

```

`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.

```

2445 \renewcommand*{\@gls@write@entrycounts}{%
2446   \immediate\write\@auxout
2447   {\string\providecommand*{\string\@gls@entry@count}[2]{}}%
2448   \count@=0\relax
2449   \forallglsentries{\@glsentry}{%
2450     \glshasattribute{\@glsentry}{entrycount}%
2451     {%
2452       \ifglused{\@glsentry}%
2453       {%
2454         \immediate\write\@auxout
2455         {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
2456       }%
2457     }%
2458     \advance\count@ by \@ne
2459   }%
2460 }%
2461 }%
2462 \ifnum\count@=0
2463   \GlossariesExtraWarningNoLine{Entry counting has been enabled
2464     \MessageBreak with \string\glsenableentrycount\space but the
2465     \MessageBreak attribute 'entrycount' hasn't
2466     \MessageBreak been assigned to any of the defined
2467     \MessageBreak entries}%
2468 \fi
2469 }

```

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

```

2470 \newcommand*{\glxtrifcounttrigger}[3]{%
2471   \glshasattribute{#1}{entrycount}%
2472   {%
2473     \ifnum\glsentryprevcount{#1}>\glsgetattribute{#1}{entrycount}\relax
2474     #3%

```

```

2475 \else
2476 #2%
2477 \fi
2478 }%
2479 {#3}%
2480 }

```

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

\@@cgl@

```

2481 \def\@@cgl@#1#2[#3]{%
2482 \glxtrifcounttrigger{#2}%
2483 {%
2484 \cglformat{#2}{#3}%
2485 \glunset{#2}%
2486 }%
2487 {%
2488 \@gls@{#1}{#2}[#3]%
2489 }%
2490 }%

```

\@@cgl@

```

2491 \def\@@cgl@#1#2[#3]{%
2492 \glxtrifcounttrigger{#2}%
2493 {%
2494 \cglformat{#2}{#3}%
2495 \glunset{#2}%
2496 }%
2497 {%
2498 \@gls@{#1}{#2}[#3]%
2499 }%
2500 }%

```

\@@cGls@

```

2501 \def\@@cGls@#1#2[#3]{%
2502 \glxtrifcounttrigger{#2}%
2503 {%
2504 \cGlsformat{#2}{#3}%
2505 \glunset{#2}%
2506 }%
2507 {%
2508 \@Gls@{#1}{#2}[#3]%
2509 }%
2510 }%

```

\@@cGlspl@

```

2511 \def\@@cGlspl@#1#2[#3]{%
2512 \glxtrifcounttrigger{#2}%
2513 {%

```

```

2514     \cGlsplformat{#2}{#3}%
2515     \glsunset{#2}%
2516 }%
2517 {%
2518     \@Glspl@{#1}{#2}[#3]%
2519 }%
2520 }%

```

\@cGLS@

```

2521 \def\@cGLS@#1#2[#3]{%
2522     \glsxtrifcounttrigger{#2}%
2523     {%
2524         \cGLSformat{#2}{#3}%
2525         \glsunset{#2}%
2526     }%
2527     {%
2528         \@GLS@{#1}{#2}[#3]%
2529     }%
2530 }%

```

\@cGLSpl@

```

2531 \def\@cGLSpl@#1#2[#3]{%
2532     \glsxtrifcounttrigger{#2}%
2533     {%
2534         \cGLSplformat{#2}{#3}%
2535         \glsunset{#2}%
2536     }%
2537     {%
2538         \@GLSpl@{#1}{#2}[#3]%
2539     }%
2540 }%

```

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

\@cgl's@

```

2541 \def\@cgl's@#1#2[#3]{\@gls@{#1}{#2}[#3]}

```

\@cGls@

```

2542 \def\@cGls@#1#2[#3]{\@Gls@{#1}{#2}[#3]}

```

\@cgl'spl@

```

2543 \def\@cgl'spl@#1#2[#3]{\@glspl@{#1}{#2}[#3]}

```

\@cGlspl@

```

2544 \def\@cGlspl@#1#2[#3]{\@Glspl@{#1}{#2}[#3]}

```

Add all upper case versions not provided by glossaries.

\cGLS

```
2545 \newrobustcmd*{\cGLS}{\@gls@hyp@opt\@cGLS}
```

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

```
2546 \newcommand*{\@cGLS}[2] [] {%
2547   \new@ifnextchar[{\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2} []}%
2548 }
```

\@cGLS@

```
2549 \def\@cGLS@#1#2[#3]{\@cGLS@{#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.

```
2550 \newcommand*{\cGLSformat}[2] {%
2551   \expandafter\mfirstuc\MakeUppercase\expandafter{\cGLSformat{#1}{#2}}%
2552 }
```

\cGLSpl

```
2553 \newrobustcmd*{\cGLSpl}{\@gls@hyp@opt\@cGLSpl}
```

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

```
2554 \newcommand*{\@cGLSpl}[2] [] {%
2555   \new@ifnextchar[{\@cGLSpl@{#1}{#2}}{\@cGLSpl@{#1}{#2} []}%
2556 }
```

\@cGLSpl@

```
2557 \def\@cGLSpl@#1#2[#3]{\@cGLSpl@{#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.

```
2558 \newcommand*{\cGLSplformat}[2] {%
2559   \expandafter\mfirstuc\MakeUppercase\expandafter{\cGLSplformat{#1}{#2}}%
2560 }
```

Modify the trigger formats to check for the regular attribute.

\cglformat

```
2561 \renewcommand*{\cglformat}[2] {%
2562   \glsifregular{#1}
2563   {\glsentryfirst{#1}}%
2564   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
2565 }
```

\cGlsformat

```
2566 \renewcommand*{\cGlsformat}[2] {%
2567   \glsifregular{#1}
2568   {\Glsentryfirst{#1}}%
2569   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
2570 }
```

\cglsplformat

```
2571 \renewcommand*{\cglsplformat}[2]{%
2572   \glsifregular{#1}
2573   {\glsentryfirstplural{#1}}%
2574   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
2575 }
```

\cGlsplformat

```
2576 \renewcommand*{\cGlsplformat}[2]{%
2577   \glsifregular{#1}
2578   {\Glsentryfirstplural{#1}}%
2579   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}}%#2%
2580 }
```

New code similar to above for unit counting.

defunitcounters

```
2581 \newcommand*{\@@newglossaryentry@defunitcounters}{%
2582   \edef\@glo@countunit{\csuse{@glxtr@categoryattr@@\@glo@category @unitcount}}%
2583   \ifdefvoid\@glo@countunit
2584   {}%
2585   {%
2586     \@glxtr@ifunitcounter{\@glo@countunit}%
2587     {}%
2588     {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}%
2589   }%
2590 }
```

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

```
2591 \newcommand*{\@glxtr@unitcountlist}{}%
```

@addunitcounter

```
2592 \newcommand*{\@glxtr@addunitcounter}[1]{%
2593   \listadd{\@glxtr@unitcountlist}{#1}%
2594   \ifcsundef{glxtr@theunit@#1}
2595   {%
2596     \ifcsdef{theH#1}%
2597     {\csdef{glxtr@theunit@#1}{\csuse{theH#1}}}%
2598     {\csdef{glxtr@theunit@#1}{\csuse{the#1}}}%
2599   }%
2600   {}%
2601 }
```

r@ifunitcounter

```
2602 \newcommand*{\@glxtr@ifunitcounter}[3]{%
2603   \xifinlist{#1}{\@glxtr@unitcountlist}{#2}{#3}%
2604 }
```

urrentunitcount

```
2605 \newcommand*\@glxtr@currentunitcount[1]{%
2606   glo@\glstoklabel{#1}@currunit@\glstgetattribute{#1}{unitcount}.%
2607   \csuse{glxtr@theunit@\glstgetattribute{#1}{unitcount}}%
2608 }
```

eviousunitcount

```
2609 \newcommand*\@glxtr@previousunitcount[1]{%
2610   glo@\glstoklabel{#1}@prevunit@\glstgetattribute{#1}{unitcount}.%
2611   \csuse{glxtr@theunit@\glstgetattribute{#1}{unitcount}}%
2612 }
```

t@currunitcount

```
2613 \newcommand*\@glst@increment@currunitcount[1]{%
2614   \glshasattribute{#1}{unitcount}%
2615   {%
2616     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
2617     \ifcsundef{\@glxtr@csname}%
2618     {%
2619       \csgdef{\@glxtr@csname}{1}%
2620       \listcsxadd
2621         {glo@\glstoklabel{#1}@unitlist}%
2622         {\glstgetattribute{#1}{unitcount}.%
2623          \csuse{glxtr@theunit@\glstgetattribute{#1}{unitcount}}%
2624         }%
2625     }%
2626     {%
2627       \csxdef{\@glxtr@csname}%
2628         {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
2629     }%
2630   }%
2631   {%
2632 }
```

t@currunitcount

```
2633 \newcommand*\@glst@local@increment@currunitcount[1]{%
2634   \glshasattribute{#1}{unitcount}%
2635   {%
2636     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
2637     \ifcsundef{\@glxtr@csname}%
2638     {%
2639       \csdef{\@glxtr@csname}{1}%
2640       \listcseadd
2641         {glo@\glstoklabel{#1}@unitlist}%
2642         {\glstgetattribute{#1}{unitcount}.%
2643          \csuse{glxtr@theunit@\glstgetattribute{#1}{unitcount}}%
2644         }%
2645     }%
2646     {%
```

```

2647 \csedef{\@glsxtr@csname}%
2648 {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
2649 }%
2650 }%
2651 {}%
2652 }

```

r@currunitcount

```

2653 \newcommand*{\@glsxtr@currunitcount}[2]{%
2654 \ifcsundef
2655 {glo@\glsdetoklabel{#1}@currunit@#2}%
2656 {0}%
2657 {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}%
2658 }%

```

r@prevunitcount

```

2659 \newcommand*{\@glsxtr@prevunitcount}[2]{%
2660 \ifcsundef
2661 {glo@\glsdetoklabel{#1}@prevunit@#2}%
2662 {0}%
2663 {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}%
2664 }%

```

eentryunitcount

```

2665 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
2666 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
2667 \renewcommand*{\gls@defdocnewglossaryentry}{%
2668 \renewcommand*\newglossaryentry[2]{%
2669 \PackageError{glossaries}{\string\newglossaryentry\space
2670 may only be used in the preamble when entry counting has
2671 been activated}{If you use \string\glsenableentryunitcount\space
2672 you must place all entry definitions in the preamble not in
2673 the document environment}%
2674 }%
2675 }%

```

New commands to access new fields:

```

2676 \newcommand*{\glsentrycurrcount}[1]{%
2677 \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2678 \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2679 }%
2680 \newcommand*{\glsentryprevcount}[1]{%
2681 \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2682 \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2683 }%

```

Access total count:

```

2684 \newcommand*{\glsentryprevtotalcount}[1]{%
2685   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
2686   {0}%
2687   {%
2688     \number\csuse{glo@\glsdetoklabel{##1}@prevunittotal}
2689   }%
2690 }%

```

Access max value:

```

2691 \newcommand*{\glsentryprevmaxcount}[1]{%
2692   \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
2693   {0}%
2694   {%
2695     \number\csuse{glo@\glsdetoklabel{##1}@prevunitmax}
2696   }%
2697 }%

```

Adjust post unset and reset:

```

2698 \let\@glsxtr@entryunitcount@org@unset\glsxtrpostunset
2699 \renewcommand*{\glsxtrpostunset}[1]{%
2700   \@glsxtr@entryunitcount@org@unset{##1}%
2701   \@gls@increment@currunitcount{##1}%
2702 }%
2703 \let\@glsxtr@entryunitcount@org@localunset\glsxtrpostlocalunset
2704 \renewcommand*{\glsxtrpostlocalunset}[1]{%
2705   \@glsxtr@entryunitcount@org@localunset{##1}%
2706   \@gls@local@increment@currunitcount{##1}%
2707 }%
2708 \let\@glsxtr@entryunitcount@org@reset\glsxtrpostreset
2709 \renewcommand*{\glsxtrpostreset}[1]{%
2710   \glshasattribute{##1}{unitcount}%
2711   {%
2712     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{##1}}%
2713     \ifcsundef{\@glsxtr@csname}%
2714     {}%
2715     {\csgdef{\@glsxtr@csname}{0}}%
2716   }%
2717   {}%
2718 }%
2719 \let\@glsxtr@entryunitcount@org@localreset\glsxtrpostlocalreset
2720 \renewcommand*{\glsxtrpostlocalreset}[1]{%
2721   \@glsxtr@entryunitcount@org@localreset{##1}%
2722   \glshasattribute{##1}{unitcount}%
2723   {%
2724     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{##1}}%
2725     \ifcsundef{\@glsxtr@csname}%
2726     {}%
2727     {\csdef{\@glsxtr@csname}{0}}%
2728   }%

```



```

2729    {}%
2730  }%

```

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

```

2731  \let\@cgls@\@cgls@
2732  \let\@cglspl@\@cglspl@
2733  \let\@cGLS@\@cGLS@
2734  \let\@cGLspl@\@cGLspl@
2735  \let\@cGLS@\@cGLS@
2736  \let\@cGLSpl@\@cGLSpl@

```

Write information to the aux file.

```

2737  \AtEndDocument{\@gls@write@entryunitcounts}%
2738  \renewcommand*{\@gls@entry@unitcount}[3]{%
2739    \csgdef{glo@glstdetoklabel{##1}@prevunit@##3}{##2}%
2740    \ifcsundef{glo@glstdetoklabel{##1}@prevunittotal}%
2741    {\csgdef{glo@glstdetoklabel{##1}@prevunittotal}{##2}}%
2742    {%
2743      \csxdef{glo@glstdetoklabel{##1}@prevunittotal}{
2744        \number\numexpr\csuse{glo@glstdetoklabel{##1}@prevunittotal}+##2}%
2745      }%
2746      \ifcsundef{glo@glstdetoklabel{##1}@prevunitmax}%
2747      {\csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}}%
2748      {%
2749        \ifnum\csuse{glo@glstdetoklabel{##1}@prevunitmax}<##2
2750        \csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}%
2751        \fi
2752      }%
2753    }%
2754  \let\glsenableentryunitcount\relax
2755  \renewcommand*{\glsenableentrycount}{%
2756    \PackageError{glossaries-extra}{\string\glsenableentrycount\space
2757      can't be used with \string\glsenableentryunitcount}%
2758    {Use one or other but not both commands}%
2759  }%
2760 }
2761 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

2762 \newcommand*{\@gls@entry@unitcount}[3]{

```

ryunitcounts@do

```

2763 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
2764   \immediate\write\@auxout
2765   {\string\@gls@entry@unitcount
2766     {\@glsentry}%
2767     {\@glsxtr@currunitcount{\@glsentry}{#1}%
2768     }%
2769     {#1}}%

```

2770 }

entryunitcounts

```

2771 \newcommand*{\@gls@write@entryunitcounts}{%
2772   \immediate\write\@auxout
2773     {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
2774   \count@=0\relax
2775   \forallglsentries{\@glsentry}{%
2776     \glshasattribute{\@glsentry}{unitcount}%
2777     {%
2778       \ifglsused{\@glsentry}%
2779       {%
2780         \forlistcsloop
2781           {\@gls@write@entryunitcounts@do}%
2782           {glo@\glsdetoklabel{\@glsentry}@unitlist}%
2783       }%
2784     }%
2785     \advance\count@ by \@ne
2786   }%
2787 }%
2788 }%
2789 \ifnum\count@=0
2790   \GlossariesExtraWarningNoLine{Entry counting has been enabled
2791     \MessageBreak with \string\glsenableentryunitcount\space but the
2792     \MessageBreak attribute ‘unitcount’ hasn’t
2793     \MessageBreak been assigned to any of the defined
2794     \MessageBreak entries}%
2795 \fi
2796 }
```

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.

```

2797 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
```

Enable entry counting:

```

2798   \glsenableentryunitcount
```

Redefine \gls etc:

```

2799   \renewcommand*{\gls}{\cgl}%
2800   \renewcommand*{\Gls}{\cGls}%
2801   \renewcommand*{\glspl}{\cglspl}%
2802   \renewcommand*{\Glspl}{\cGlspl}%
2803   \renewcommand*{\GLS}{\cGLS}%
2804   \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```

2805   \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%
```

In case this command is used again:

```

2806   \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
2807   \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

```

2808 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
2809   can't be used with \string\GlsXtrEnableEntryUnitCounting}%
2810 {Use one or other but not both commands}}%
2811 }

```

countunsetattr

```

2812 \newcommand*{\@glstr@setentryunitcountunsetattr}[3]{%
2813   \@for\@glstr@cat:=#1\do
2814   {%
2815     \ifdefempty{\@glstr@cat}{}%
2816     {%
2817       \glsssetcategoryattribute{\@glstr@cat}{entrycount}{#2}%
2818       \glsssetcategoryattribute{\@glstr@cat}{unitcount}{#3}%
2819     }%
2820   }%
2821 }

```

1.3.6 Acronym Modifications

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

nericNewAcronym

```

2822 \renewcommand*{\SetGenericNewAcronym}{%
2823   \let\@Gls@entryname\@Gls@acronymname
2824   \renewcommand{\newacronym}[4][{}]{%
2825     \ifdefempty{\@glsacronymlists}%
2826     {%
2827       \def\@glo@type{\acronymtype}%
2828       \setkeys{glossentry}{##1}%
2829       \DeclareAcronymList{\@glo@type}%
2830     }%
2831   }%
2832   \glskeylisttok{##1}%
2833   \glslabeltok{##2}%
2834   \glsshorttok{##3}%
2835   \glslongtok{##4}%
2836   \newacronymhook
2837   \protected@edef\@do@newglossaryentry{%
2838     \noexpand\newglossaryentry{\the\glslabeltok}%
2839     {%
2840       type=\acronymtype,%
2841       name={\expandonce{\acronymentry{##2}}},%
2842       sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%

```

```

2843     text={\the\glsshorttok},%
2844     short={\the\glsshorttok},%
2845     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
2846     long={\the\glslongtok},%
2847     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
2848     category=acronym,
2849     \GenericAcronymFields,%
2850     \the\glskeylisttok
2851   }%
2852 }%
2853 \do@newglossaryentry
2854 }%
2855 \renewcommand*{\acrfullfmt}[3]{%
2856   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
2857 \renewcommand*{\Acrfullfmt}[3]{%
2858   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
2859 \renewcommand*{\ACRfullfmt}[3]{%
2860   \glslink[##1]{##2}{%
2861     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
2862 \renewcommand*{\acrfullplfmt}[3]{%
2863   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
2864 \renewcommand*{\Acrfullplfmt}[3]{%
2865   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
2866 \renewcommand*{\ACRfullplfmt}[3]{%
2867   \glslink[##1]{##2}{%
2868     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
2869 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}}%
2870 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
2871 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}}%
2872 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
2873 }

```

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:

```

2874 \let\@glstr@org@setacronymstyle\setacronymstyle
2875 \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`.

```

2876 \newcommand*{\MakeAcronymsAbbreviations}{%
2877   \renewcommand*{\newacronym}[4][{}]{%
2878     \glstr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
2879   }%
2880 \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
2881 \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%
2882 \renewcommand*{\setacronymstyle}[1]{%
2883   \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}

```

```

2884     unavailable.
2885     Use \string\setabbreviationstyle\space instead.
2886     The original acronym interface can be restored with
2887     \string\RestoreAcronyms}{}%
2888 }%
2889 \renewcommand*\newacronymstyle}[1]{%
2890     \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
2891     available unless you restore the original acronym interface with
2892     \string\RestoreAcronyms}%
2893     \@glstr@org@newacronymstyle{##1}%
2894 }%
2895 }

```

Switch acronyms to abbreviations:

```
2896 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```

2897 \newcommand*\RestoreAcronyms{%
2898     \SetGenericNewAcronym
2899     \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
2900     \renewcommand{\acronymfont}[1]{##1}%
2901     \let\setacronymstyle\@glstr@org@setacronymstyle
2902     \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.

```

2903     \renewcommand*\@gls@link@checkfirsthyper{%
2904         \ifglsused{\glslabel}%
2905         {\let\glstrifwasfirstuse\@secondoftwo}
2906         {\let\glstrifwasfirstuse\@firstoftwo}%
2907         \@glstr@org@checkfirsthyper
2908     }
2909     \glssetcategoryattribute{acronym}{regular}{false}%
2910     \setacronymstyle{long-short}%
2911 }

```

\glsacspace Allow the user to customise the maximum value.

```

2912 \renewcommand*\glsacspace}[1]{%
2913     \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
2914     \ifdim\dimen@<\glsacspacemax~\else\space\fi
2915 }

```

\glsacspacemax Value used in the above.

```
2916 \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

```
2917 \newcommand*{\@glxtr@reg@glosslist}{}
```

Save the original definition of `\makeglossaries`:

```
2918 \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.

\makeglossaries

```
2919 \renewcommand*{\makeglossaries}[1][]{%
2920   \ifblank{#1}%
2921   {\@glxtr@org@makeglossaries}%
2922   {%
2923     \edef\@glxtr@reg@glosslist{#1}%
2924     \ifundef{\glswrite}{\newwrite\glswrite}{}%
2925     \protected@write\@auxout{}\string\providecommand
2926       \string\@glorder[1]{}%
2927     \protected@write\@auxout{}\string\providecommand
2928       \string\@istfilename[1]{}%
2929     \protected@write\@auxout{}\string\@istfilename{\istfilename}%
2930     \protected@write\@auxout{}\string\@glorder{\glorder}%
2931     \protected@write\@auxout{}\string\glxtr@makeglossaries{#1}%
2932     \write\@auxout{\string\providecommand\string\@glreference[3]{}}%

```

Iterate through each supplied glossary type and activate it.

```
2933   \@for\@glo@type:=#1\do{%
2934     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
2935   }%
```

New glossaries must be created before `\makeglossaries`:

```
2936 \renewcommand*{\newglossary}[4][]{%
2937   \PackageError{glossaries}{New glossaries
2938     must be created before \string\makeglossaries}{You need
2939     to move \string\makeglossaries\space after all your
2940     \string\newglossary\space commands}}%
```

Any subsequence instances of this command should have no effect

```
2941 \let\@makeglossary\relax
2942 \let\makeglossary\relax
2943 \renewcommand\makeglossaries[1][]{}
```

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

```
2944 \@disable@onlypremakeg
```

Allow see key:

```
2945 \let\glsc@checkseeallowed\relax
```

Adjust \@do@seeglossary

```

2946 \let\@glxtr@org@doseeglossary\@do@seeglossary
2947 \renewcommand*{\@do@seeglossary}[2]{%
2948   \edef\@gls@label{\glstoklabel{##1}}%
2949   \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
2950   \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glxtr@reg@glosslist}%
2951   {\@glxtr@org@doseeglossary{##1}{##2}}%
2952   {%
2953     \protected@write\@auxout{}{%
2954       \string\@gls@reference
2955         {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
2956     }%
2957   }%
2958 }%

```

Adjust @@do@@wrglossary

```

2959 \let\@glxtr@@do@@wrglossary@@do@@wrglossary
2960 \def@@do@@wrglossary{%
2961   \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
2962   \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glxtr@reg@glosslist}%
2963   {\@glxtr@@do@@wrglossary}%
2964   {\gls@noidxglossary}%
2965 }%

```

Suppress warning about no \makeglossaries

```

2966 \let\warn@nomakeglossaries\relax
2967 \def\warn@noprntglossary{%
2968   \GlossariesWarningNoLine{No \string\printglossary\space
2969     or \string\printglossaries\space
2970     found.^^J(Remove \string\makeglossaries\space if you don't want
2971     any glossaries.)^^JThis document will not have a glossary}%
2972 }%

```

Only warn for glossaries not listed.

```

2973 \renewcommand{\@gls@noref@warn}[1]{%
2974   \edef\@gls@type{##1}%
2975   \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glxtr@reg@glosslist}%
2976   {%
2977     \GlossariesExtraWarning{Can't use
2978       \string\printnoidxglossary[type={\@gls@type}]
2979       when '\@gls@type' is listed in the optional argument of
2980       \string\makeglossaries}%
2981   }%
2982   {%
2983     \GlossariesWarning{Empty glossary for
2984       \string\printnoidxglossary[type={##1}].
2985       Rerun may be required (or you may have forgotten to use
2986       commands like \string\gls)}%
2987   }%
2988 }%

```

Adjust display number list to check for type:

```
2989 \renewcommand*{\glsdisplaynumberlist}[1]{%
2990   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2991   {\@glsxtr@idx@displaynumberlist{##1}}%
2992   {\@glsxtr@noidx@displaynumberlist{##1}}%
2993 }%
```

Adjust entry list:

```
2994 \renewcommand*{\glsentrynumberlist}[1]{%
2995   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2996   {\@glsxtr@idx@entrynumberlist{##1}}%
2997   {\@glsxtr@noidx@entrynumberlist{##1}}%
2998 }%
```

Adjust number list loop

```
2999 \renewcommand*{\glsnumberlistloop}[2]{%
3000   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3001   {%
3002     \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3003       not available for glossary ‘##1’}{}%
3004   }%
3005   {\@glsxtr@noidx@numberlistloop{##1}{##2}}%
3006 }%
```

Only sanitize sort for normal indexing glossaries.

```
3007 \renewcommand*{\glsprestandardsort}[3]{%
3008   \expandafter\DTLifinlist\expandafter{##2}{\@glsxtr@reg@glosslist}%
3009   {%
3010     \glsdosanitizesort
3011   }%
3012   {%
3013     \ifglssanitizesort
3014       \@gls@noidx@sanitizesort
3015     \else
3016       \@gls@noidx@nosanitizesort
3017     \fi
3018   }%
3019 }%
```

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

```
3020 \renewcommand*{\new@glossaryentry}[2]{%
3021   \PackageError{glossaries-extra}{Glossary entries must be defined
3022     in the preamble\MessageBreak when you use the optional argument
3023     of \string\makeglossaries}{Either move your definitions to the
3024     preamble or don't use the optional argument of
3025     \string\makeglossaries}%
3026 }%
```

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).


```

3027 \let\@glo@assign@sortkey\@glxtr@mixed@assign@sortkey
3028 \renewcommand*{\@printgloss@setsort}{%

```

Need to extract just the type value.

```

3029 \expandafter\@glxtr@gettype\expandafter,\@glxtr@printglossopts,%
3030 type=\glsdefaulttype,\@end@glxtr@gettype
3031 \def\@glo@sorttype{\@glo@default@sorttype}%
3032 }%

```

Check automake setting:

```

3033 \ifglsautomake
3034 \renewcommand*{\@gls@doautomake}{%
3035 \@for\@gls@type:=\@glxtr@reg@glosslist\do{%
3036 \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3037 }%
3038 }%
3039 \fi
3040 }%
3041 }

```

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

`rgprintglossary` This no longer simply saves `\@printglossary` with `\let` 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.)

```

3042 \newcommand{\@glxtr@orgprintglossary}[2]{%
3043 \def\@glo@type{\glsdefaulttype}%

```

Add check here.

```

3044 \def\glossarytitle{%
3045 \ifcsdef{\@glotype@\@glo@type @title}%
3046 {\csuse{\@glotype@\@glo@type @title}}%
3047 {\glossaryname}}%
3048 \def\glossarytoctitle{\glossarytitle}%
3049 \let\org@glossarytitle\glossarytitle
3050 \def\@glossarystyle{%
3051 \ifx\@glossary@default@style\relax
3052 \GlossariesWarning{No default glossary style provided \MessageBreak
3053 for the glossary '\@glo@type'. \MessageBreak
3054 Using deprecated fallback. \MessageBreak
3055 To fix this set the style with \MessageBreak
3056 \string\setglossarystyle\space or use the \MessageBreak
3057 style key=value option}%
3058 \fi
3059 }%
3060 \def\gls@dotoc\toctitle{\glssettoc\toc\toctitle{\@glo@type}}%
3061 \let\@org@glossaryentrynumbers\glossaryentrynumbers
3062 \bgroup

```

```

3063 \printgloss@setsort
3064 \setkeys{printgloss}{#1}%
3065 \ifx\glossarytitle\org@glossarytitle
3066 \else
3067 \cslet{@glotype@\@glo@type @title}{\glossarytitle}%
3068 \fi
3069 \let\currentglossary\@glo@type
3070 \let\org@glossaryentrynumbers\glossaryentrynumbers
3071 \let\glsnonextpages\@glsnonextpages
3072 \let\glsnextpages\@glsnextpages
3073 \let\nopostdesc\@nopostdesc
3074 \gls@dotoc@title
3075 \@glossarystyle
3076 \let\gls@org@glossaryentryfield\glossentry
3077 \let\gls@org@glossarysubentryfield\subglossentry
3078 \renewcommand{\glossentry}[1]{%
3079 \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3080 \gls@org@glossaryentryfield{##1}%
3081 }%
3082 \renewcommand{\subglossentry}[2]{%
3083 \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3084 \gls@org@glossarysubentryfield{##1}{##2}%
3085 }%
3086 \@gls@preglossaryhook
3087 #2%
3088 \egroup
3089 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
3090 \global\let\warn@noprintglossary\relax
3091 }

```

\printglossary Redefine.

```

3092 \renewcommand{\@printglossary}[2]{%
3093 \def\@glsxtr@printglossopts{#1}%
3094 \@glsxtr@orgprintglossary{#1}{#2}%
3095 }

```

Add a key that switches off the entry targets:

```

3096 \define@choicekey{printgloss}{target}[\val\nr]{true,false}[true]{%
3097 \ifcase\nr
3098 \let\@glstarget\glsdohypertarget
3099 \else
3100 \let\@glstarget\@secondoftwo
3101 \fi
3102 }

```

@makeglossaries For the benefit of makeglossaries

```

3103 \newcommand*{\glsxtr@makeglossaries}[1]{%

```

@glsxtr@gettype Get just the type.

```

3104 \def\@glsxtr@gettype#1,type=#2,#3\@end@glsxtr@gettype{%
3105   \def\@glo@type{#2}%
3106 }

```

@assign@sortkey Assign the sort key.

```

3107 \newcommand\@glsxtr@mixed@assign@sortkey[1]{%
3108   \edef\@glo@type{\@glo@type}%
3109   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glsxtr@reg@glosslist}%
3110   {%
3111     \@glo@no@assign@sortkey{#1}%
3112   }%
3113   {%
3114     \@glo@assign@sortkey{#1}%
3115   }%
3116 }%

```

Display number list for the regular version:

splaynumberlist

```

3117 \let\@glsxtr@idx@displaynumberlist\glsdisplaynumberlist

```

Display number list for the “noidx” version:

splaynumberlist

```

3118 \newcommand*{\@glsxtr@noidx@displaynumberlist}[1]{%
3119   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3120   \ifdef\@gls@loclist
3121   {%
3122     \def\@gls@noidxloclist@sep{%
3123       \def\@gls@noidxloclist@sep{%
3124         \def\@gls@noidxloclist@sep{%
3125           \glsnumlistsep
3126         }%
3127       \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
3128     }%
3129   }%
3130   \def\@gls@noidxloclist@finalsep{}%
3131   \def\@gls@noidxloclist@prev{}%
3132   \forlistloop{\@glsnoidxdisplayloclisthandler}{\@gls@loclist}%
3133   \@gls@noidxloclist@finalsep
3134   \@gls@noidxloclist@prev
3135 }%
3136 {%
3137   ??\glsdoifexists{#1}%
3138   {%
3139     \GlossariesWarning{Missing location list for ‘#1’. Either
3140       a rerun is required or you haven’t referenced the entry.}%
3141   }%
3142 }%
3143 }%

```

3144

And for the number list loop:

@numberlistloop

```
3145 \newcommand*{\@glstr@noidx@numberlistloop}[3]{%
3146   \letcs{\@gl@loclist}{glo@\glstoklabel{#1}@loclist}%
3147   \let\@gl@org@glsnoidxdisplayloc\glsnoidxdisplayloc
3148   \let\@gl@org@glseeformat\glseeformat
3149   \let\glsnoidxdisplayloc#2\relax
3150   \let\glseeformat#3\relax
3151   \ifdef\@gl@loclist
3152   {%
3153     \forlistloop{\glsnoidxnumberlistloophandler}{\@gl@loclist}%
3154   }%
3155   {%
3156     ??\glstoifexists{#1}%
3157     {%
3158       \GlossariesWarning{Missing location list for ‘##1’. Either
3159         a rerun is required or you haven’t referenced the entry.}%
3160     }%
3161   }%
3162   \let\glsnoidxdisplayloc\@gl@org@glsnoidxdisplayloc
3163   \let\glseeformat\@gl@org@glseeformat
3164 }%
```

Same for entry number list.

entrynumberlist

```
3165 \newcommand*{\@glstr@noidx@entrynumberlist}[1]{%
3166   \letcs{\@gl@loclist}{glo@\glstoklabel{#1}@loclist}%
3167   \ifdef\@gl@loclist
3168   {%
3169     \glsnoidxloclist{\@gl@loclist}%
3170   }%
3171   {%
3172     ??\glstoifexists{#1}%
3173     {%
3174       \GlossariesWarning{Missing location list for ‘#1’. Either
3175         a rerun is required or you haven’t referenced the entry.}%
3176     }%
3177   }%
3178 }%
```

entrynumberlist

```
3179 \newcommand*{\@glstr@idx@entrynumberlist}[1]{\glstrynumberlist{#1}}
```

@noidx@glossary

```
3180 \renewcommand*{\@print@noidx@glossary}{%
3181   \ifcsdef{\gl@ref@{\gl@type}}%
```

```

3182 {%
3183   \ifcsdef{@glo@sortmacro@\@glo@sorttype}%
3184   {%
3185     \csuse{@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
3186   }%
3187   {%
3188     \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
3189   }%
3190   \glossarysection[\glossarytoctitle]{\glossarytitle}%
3191   \glossarypreamble

```

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

```

3192   \def\@gls@currentlettergroup{}%
3193   \begin{theglossary}%
3194   \glossaryheader
3195   \glsresetentrylist
3196   \forlistcsloop{\@gls@noidx@do}{\@glsref@\@glo@type}%
3197   \end{theglossary}%
3198   \glossarypostamble
3199 }%
3200 {%

```

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

```

3201   \glsxtrifemptyglossary{\@glo@type}%
3202   {}%
3203   {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
3204   \@gls@noref@warn{\@glo@type}%
3205 }%
3206 }

```

`noidxdisplayloc` Patch to check for range formations.

```

3207 \renewcommand*{\glsnoidxdisplayloc}[4]{%
3208   \setentrycounter[#1]{#2}%
3209   \@glsxtr@display@loc#3\empty\end@glxtr@display@loc{#4}%
3210 }

```

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

```

3211 \def\@glsxtr@display@loc#1#2\end@glxtr@display@loc#3{%
3212   \ifx#1(\relax
3213     \glsxtrdisplaystartloc{#2}{#3}%
3214   \else
3215     \ifx#1)\relax
3216       \glsxtrdisplayendloc{#2}{#3}%
3217     \else
3218       \glsxtrdisplaysingleloc{#1#2}{#3}%
3219     \fi
3220   \fi
3221 }

```

`\displaysingleloc` Single location.

```
3222 \newcommand*{\glxtrdisplaysingleloc}[2]{%
3223   \csuse{#1}{#2}%
3224 }
```

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.

```
3225 \newcommand*{\glxtrdisplaystartloc}[2]{%
3226   \edef\glxtrlocrangefmt{#1}%
3227   \ifx\glxtrlocrangefmt\empty
3228     \def\glxtrlocrangefmt{\glsnnumberformat}%
3229   \fi
3230   \expandafter\glxtrdisplaysingleloc
3231   \expandafter{\glxtrlocrangefmt}{#2}%
3232 }
```

`\trdisplayendloc` End of a location range.

```
3233 \newcommand*{\glxtrdisplayendloc}[2]{%
3234   \ifdefstring{\glxtrlocrangefmt}{#1}{}%
3235   {\GlossariesExtraWarning{Mismatched end location range
3236     (start=\glxtrlocrangefmt, end=#1)}}%
3237   }%
3238   \glxtrdisplayendloohook{#1}{#2}%
3239   \expandafter\glxtrdisplaysingleloc
3240   \expandafter{\glxtrlocrangefmt}{#2}%
3241   \def\glxtrlocrangefmt{}%
3242 }
```

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

```
3243 \newcommand*{\glxtrdisplayendloohook}[2]{}
```

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

```
3244 \newcommand*{\glxtrlocrangefmt}{}%
```

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

`@print@glossary`

```
3245 \renewcommand{\@print@glossary}{%
3246   \makeatletter
3247   \@input@{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
3248   \IfFileExists{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
3249   {}%
3250   {\glxtrNoGlossaryWarning{\@glo@type}}%
3251   \ifglxindy
3252     \ifcsundef{@xdy@\@glo@type @language}%

```

```

3253 {%
3254     \edef\@do@auxoutstuff{%
3255         \noexpand\AtEndDocument{%
3256             \noexpand\immediate\noexpand\write\@auxout{%
3257                 \string\providecommand\string\@xdylanguage[2]{}}%
3258             \noexpand\immediate\noexpand\write\@auxout{%
3259                 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
3260         }%
3261     }%
3262 }%
3263 {%
3264     \edef\@do@auxoutstuff{%
3265         \noexpand\AtEndDocument{%
3266             \noexpand\immediate\noexpand\write\@auxout{%
3267                 \string\providecommand\string\@xdylanguage[2]{}}%
3268             \noexpand\immediate\noexpand\write\@auxout{%
3269                 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
3270                     @language\endcsname}}%
3271         }%
3272     }%
3273 }%
3274 \@do@auxoutstuff
3275 \edef\@do@auxoutstuff{%
3276     \noexpand\AtEndDocument{%
3277         \noexpand\immediate\noexpand\write\@auxout{%
3278             \string\providecommand\string\@gls@codepage[2]{}}%
3279         \noexpand\immediate\noexpand\write\@auxout{%
3280             \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
3281     }%
3282 }%
3283 \@do@auxoutstuff
3284 \fi
3285 \renewcommand*{\@warn@nomakeglossaries}{%
3286     \GlossariesWarningNoLine{\string\makeglossaries\space
3287         hasn't been used,^^Jthe glossaries will not be updated}%
3288 }%
3289 }

```

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

`oGlsWarningHead` Header message.

```

3290 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
3291     This document is incomplete. The external file associated with
3292     the glossary '#1' (which should be called \texttt{#2})
3293     hasn't been created.%
3294 }

```

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

```

3295 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%

```

```

3296 This has probably happened because there are no entries defined
3297 in this glossary.%
3298 }

```

arningEmptyMain The default “main” glossary is empty.

```

3299 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
3300 If you don't want this glossary,
3301 add \texttt{nomain} to your package option list when you load
3302 \texttt{glossaries-extra.sty}. For example:%
3303 }

```

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

```

3304 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
3305 Did you forget to use \texttt{type=#1} when you defined your
3306 entries? If you tried to load entries into this glossary with
3307 \texttt{\string\loadglsentries} did you remember to use
3308 \texttt{[#1]} as the optional argument? If you did, check that
3309 the definitions in the file you loaded all had the type set
3310 to \texttt{\string\glsdefaulttype}.%
3311 }

```

arningCheckFile Advisory message to check the file contents.

```

3312 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
3313 Check the contents of the file \texttt{#1}. If
3314 it's empty, that means you haven't indexed any of your entries in this
3315 glossary (using commands like \texttt{\string\gls} or
3316 \texttt{\string\glsadd}) so this list can't be generated.
3317 If the file isn't empty, the document build process hasn't been
3318 completed.%
3319 }

```

WarningAutoMake Message when automake option has been used.

```

3320 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
3321 You may need to rerun \LaTeX. If you already have, it may be that
3322 \TeX's shell escape doesn't allow you to run
3323 \ifglxindy xindy\else makeindex\fi. Check the
3324 transcript file \texttt{\jobname.log}. If the shell escape is
3325 disabled, try one of the following:
3326
3327 \begin{itemize}
3328 \item Run the external (Lua) application:
3329
3330 \texttt{makeglossaries-lite.lua \string"\jobname\string"}
3331
3332 \item Run the external (Perl) application:
3333
3334 \texttt{makeglossaries \string"\jobname\string"}
3335 \end{itemize}
3336

```



```

3337 Then rerun \LaTeX\ on this document.
3338 \GlossariesExtraWarning{Rerun required to build the
3339 glossary '#1' or check TeX's shell escape allows
3340 you to run \ifglxindy xindy\else makeindex\fi}%
3341 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

3342 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
3343   You need to either replace \texttt{\string\makenoidxglossaries}
3344   with \texttt{\string\makeglossaries} or replace
3345   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
3346   \texttt{\string\printnoidxglossary}
3347   (or \texttt{\string\printnoidxglossaries}) and then rebuild
3348   this document.%
3349 }

```

WarningBuildInfo Build advice.

```

3350 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
3351   Try one of the following:
3352   \begin{itemize}
3353     \item Add \texttt{automake} to your package option list when you load
3354       \texttt{glossaries-extra.sty}. For example:
3355
3356       \texttt{\string\usepackage[automake]%
3357         \glsopenbrace glossaries-extra\glsclosebrace}
3358
3359     \item Run the external (Lua) application:
3360
3361     \texttt{makeglossaries-lite.lua \string"\jobname\string"}
3362
3363     \item Run the external (Perl) application:
3364
3365     \texttt{makeglossaries \string"\jobname\string"}
3366   \end{itemize}
3367
3368   Then rerun \LaTeX\ on this document.%
3369 }

```

oGlsWarningTail Final paragraph.

```

3370 \newcommand{\GlsXtrNoGlsWarningTail}{%
3371   This message will be removed once the problem has been fixed.%
3372 }

```

GlsWarningNoOut No out file created. Build advice.

```

3373 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
3374   The file \texttt{#1} doesn't exist. This most likely means you haven't used
3375   \texttt{\string\makeglossaries} or you have used
3376   \texttt{\string\nofiles}. If this is just a draft version of the

```

```

3377 document, you can suppress this message using the
3378 \texttt{nomissingglstext} package option.%
3379 }

```

glossarywarning

```

3380 \newcommand*{\@glstr@defaultnoglossarywarning}[1]{%
3381 \glossarysection[\glossarytoctitle]{\glossarytitle}
3382 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
3383 \par
3384 \glstrifemptyglossary{#1}%
3385 {%
3386 \GlsXtrNoGlsWarningEmptyStart\space
3387 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
3388 \medskip
3389 \noindent\texttt{\string\usepackage[nomain\ifglstacronym ,acronym\fi]%
3390 \glstopenbrace glossaries-extra\glstclosebrace}
3391 \medskip
3392 }%
3393 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
3394 }%
3395 {%
3396 \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
3397 {%
3398 \GlsXtrNoGlsWarningCheckFile
3399 {\jobname.\csname @glotype@\@glo@type @out\endcsname}
3400
3401 \ifglstautomake
3402
3403 \GlsXtrNoGlsWarningAutoMake{#1}
3404
3405 \else
3406
3407 \ifthenelse{\equal{#1}{main}}%
3408 {%
3409 \GlsXtrNoGlsWarningEmptyMain\par
3410 \medskip
3411 \noindent\texttt{\string\usepackage[nomain]%
3412 \glstopenbrace glossaries-extra\glstclosebrace}
3413 \medskip
3414 }%
3415 {}%
3416
3417 \ifdequal\makeglossaries\@no@makeglossaries
3418 {%
3419 \GlsXtrNoGlsWarningMisMatch
3420 }%
3421 {%
3422 \GlsXtrNoGlsWarningBuildInfo
3423 }%

```

```

3424     \fi
3425   }%
3426   {%
3427     \GlsXtrNoGlsWarningNoOut
3428     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
3429   }%
3430 }%
3431 \par
3432 \GlsXtrNoGlsWarningTail
3433 }

```

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

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

```

3434 \newcommand*{\glxtrresourcefile}[2] [] {%
3435   \protected@write\@auxout{}\string\glxtr@resource{#1}{#2}}%
3436   \glxtr@writefields
3437   \let\@glxtr@org@see@noindex\@gl@see@noindex
3438   \let\@gl@see@noindex\relax
3439   \IfFileExists{#2.glstex}%
3440   {%

```

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

```

3441     \edef\@bibgls@restreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
3442     \makeatletter
3443     \@input{#2.glstex}%
3444     \@bibgls@restreat
3445   }%
3446   {%
3447     \GlossariesExtraWarning{No file '#2.glstex'}%
3448   }%
3449   \let\@gl@see@noindex\@glxtr@org@see@noindex
3450 }
3451 \@onlypreamble\glxtrresourcefile

```

trresourcecount

```

3452 \newcount\glxtrresourcecount

```

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

```

3453 \newcommand*{\GlsXtrLoadResources}[1] [] {%
3454   \ifnum\glxtrresourcecount=0\relax
3455     \glxtrresourcefile[#1]{\jobname}%
3456   \else
3457     \glxtrresourcefile[#1]{\jobname-\the\glxtrresourcecount}%
3458   \fi
3459   \advance\glxtrresourcecount by 1\relax
3460 }

```

```

glxtr@resource
3461 \newcommand*{\glxtr@resource}[2]{}

\glxtr@fields
3462 \newcommand*{\glxtr@fields}[1]{}

xtr@texencoding
3463 \newcommand*{\glxtr@texencoding}[1]{}

\glxtr@langtag
3464 \newcommand*{\glxtr@langtag}[1]{}

@pluralsuffixes
3465 \newcommand*{\glxtr@pluralsuffixes}[4]{}

tr@shortcutsval
3466 \newcommand*{\glxtr@shortcutsval}[1]{}

sxtr@linkprefix
3467 \newcommand*{\glxtr@linkprefix}[1]{}

xtr@writefields  This information only needs to be written once, so disable it after it's been used.
3468 \newcommand*{\glxtr@writefields}{%
3469   \protected@write\@auxout{}\{\string\glxtr@fields{\@gls@keymap}}%
   If any languages have been loaded, the language tag will be available in \CurrentTrackedLanguageTag
   (provided by tracklang). For multilingual documents, the required locale will have to be indi-
   cated in the sort key when using \glxtrresourcefile.
3470   \ifdef\CurrentTrackedLanguageTag
3471   {%
3472     \protected@write\@auxout{}\{%
3473       \string\glxtr@langtag{\CurrentTrackedLanguageTag}}%
3474   }%
3475   {%
3476     \protected@write\@auxout{}\{\string\glxtr@pluralsuffixes
3477       {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
3478       {\glxtrabbrvpluralsuffix}}%
3479     \ifdef\inputencodingname
3480     {%
3481       \protected@write\@auxout{}\{\string\glxtr@texencoding{\inputencodingname}}%
3482     }%
3483     {%
   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.)
3484     \@ifpackageloaded{fontspec}%
3485     {\protected@write\@auxout{}\{\string\glxtr@texencoding{utf8}}}%
3486     {%
3487   }%
3488   \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.

```
3489 \AtBeginDocument
3490   {\protected@write\@auxout{}\string\glstr@linkprefix{\glolinkprefix}}}%
3491   \let\glstr@writefields\relax
3492 }
```

Allow locations specific to a particular counter to be recorded.

\glstr@record

```
3493 \newcommand*\glstr@record[5]{}
```

r@counterrecord Aux file command.

```
3494 \newcommand*\glstr@counterrecord[3]{%
3495   \glstrfieldlistgadd{#1}{record.#2}{#3}%
3496 }
```

unterrecordhook Hook used by \glstr@dorecord.

```
3497 \newcommand*\glstr@counterrecordhook{}
```

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

```
3498 \newcommand*\GlsXtrRecordCounter[1]{%
3499   \@glstr@recordcounter{#1}%
3500 }
3501 \@onlypreamble\GlsXtrRecordCounter
```

docounterrecord

```
3502 \newcommand*\glstr@docounterrecord[1]{%
3503   \protected@write\@auxout{}\string\glstr@counterrecord
3504     {\glslabel}{#1}{\csuse{the#1}}}%
3505 }
```

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

```
3506 \newcommand*\printunsrtglossary{%
3507   \@ifstar\s@printunsrtglossary\@printunsrtglossary
3508 }
```

ntunsrtglossary Unstarred version.

```
3509 \newcommand*\@printunsrtglossary[1][{}]{%
3510   \@printglossary{type=\glstypedefaulttype,#1}{\@print@unsrt@glossary}%
3511 }
```

ntunsrtglossary Starred version.

```
3512 \newcommand*\s@printunsrtglossary[2][{}]{%
3513   \begingroup
3514     #2%
3515     \@printglossary{type=\glstypedefaulttype,#1}{\@print@unsrt@glossary}%
3516   \endgroup
3517 }
```

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

```
3518 \newcommand*{\printunsrtglossaries}{%
3519   \foralllglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
3520 }
```

`@unsrt@glossary`

```
3521 \newcommand*{\@print@unsrt@glossary}{%
3522   \glossarysection[\glossarytoctitle]{\glossarytitle}%
3523   \glossarypreamble
3524   check for empty list
3525   \glstrifemptyglossary{\@glo@type}%
3526   {%
3527     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
3528   }%
3529   \key@ifundefined{glossentry}{group}%
3530   {\let\@gls@getgrouptitle\@glstr@noidx@getgrouptitle}%
3531   {\let\@gls@getgrouptitle\@glstr@unsrt@getgrouptitle}%
3532   \begin{theglossary}%
3533   \glossaryheader
3534   \glsresetentrylist
3535   \def\@gls@currentlettergroup{%
3536     \expandafter\@for\expandafter\glscurrententrylabel\expandafter
3537       :\expandafter=\csname glo@list@\@glo@type\endcsname\do{%
3538       \ifdefempty{\glscurrententrylabel}
3539       }%
3540       {\printunsrtglossaryhandler\glscurrententrylabel}%
3541     }%
3542   \end{theglossary}%
3543   }%
3544   \glossarypostamble
3545 }
```

`glossaryhandler`

```
3546 \newcommand{\printunsrtglossaryhandler}[1]{%
3547   \glstr@unsrt@do{#1}%
3548 }
```

`srtglossaryunit`

```
3549 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
3550   \s@printunsrtglossary[type=\glsdefaulttype,#1]{%
3551     \printunsrtglossaryunitsetup{#2}%
3552   }%
3553 }
```

`ossaryunitsetup`

```
3554 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
```

```

3555 \renewcommand{\printunsrtglossaryhandler}[1]{%
3556   \glstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
3557   {\glstrunsrtdo{##1}}%
3558   }%
3559 }%
3560 \ifcsundef{theH#1}%
3561 {%
3562   \renewcommand*\glolinkprefix{record.#1.\csuse{the#1}.}%
3563 }%
3564 {%
3565   \renewcommand*\glolinkprefix{record.#1.\csuse{theH#1}.}%
3566 }%
3567 \renewcommand*\glossarysection[2][{}]{%
3568 \appto\glossarypostamble{\glspare\medskip\glspare}%
3569 }

```

srtglossaryunit

```

3570 \newcommand{\print@noop@unsrtglossaryunit}[2][{}]{%
3571   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
3572     requires the record=only or record=alsoindex package option}{}%
3573 }

```

t@getgrouptitle

```

3574 \newcommand*\@glstr@unsrt@getgrouptitle[2]{%
3575   \def#2{#1}%
3576 }

```

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

```

3577 \newcommand{\glstrunsrtdo}{\@glstr@noidx@do}

```

\glstr@noidx@do Minor modification of \@gls@noidx@do to check for location field if present.

```

3578 \newcommand{\@glstr@noidx@do}[1]{%
3579   \global\letcs{\@gls@loclist}{glo@\glstdetoklabel{#1}@loclist}%
3580   \global\letcs{\@gls@location}{glo@\glstdetoklabel{#1}@location}%
3581   \ifglshasparent{#1}%
3582   {%
3583     \gls@level=\csuse{glo@\glstdetoklabel{#1}@level}\relax
3584     \ifdefvoid{\@gls@location}%
3585     {%
3586       \ifdefvoid{\@gls@loclist}%
3587       {%
3588         \subglossentry{\gls@level}{#1}{}%
3589       }%
3590     }%
3591     \subglossentry{\gls@level}{#1}%
3592     {%
3593       \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
3594     }%
3595   }%

```

```

3596 }%
3597 {%
3598   \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
3599 }%
3600 }%
3601 {%
3602   \letcs{\@gls@sort}{glo@glstdetoklabel{#1}@sort}%
3603   \key@ifundefined{glossentry}{group}%
3604   {%
3605     \expandafter\glo@grabfirst\@gls@sort{}}\@nil
3606   }%
3607   {%
3608     \protected@xdef\@glo@thislettergrp{%
3609       \csname glo@glstdetoklabel{#1}@group\endcsname}%
3610     }%
3611     \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
3612     {}%
3613     {%
3614       \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
3615       \gls@groupheading{\@glo@thislettergrp}%
3616     }%
3617     \let\@gls@currentlettergroup\@glo@thislettergrp
3618     \ifdefvoid{\@gls@location}%
3619     {%
3620       \ifdefvoid{\@gls@loclist}
3621       {%
3622         \glossentry{#1}{}%
3623       }%
3624       {%
3625         \glossentry{#1}%
3626         {%
3627           \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
3628         }%
3629       }%
3630     }%
3631     {%
3632       \glossentry{#1}%
3633     }%
3634     \glossaryentrynumbers{\@gls@location}%
3635   }%
3636 }%
3637 }%
3638 }

```

1.4 Integration with glossaries-accsupp

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

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

```
3639 \ifpackageloaded{glossaries-accsupp}
3640 {
```

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

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

```
3641 \newcommand*{\glsaccessname}[1]{%
3642 \glsnameaccessdisplay
3643 {%
3644 \glsentryname{#1}%
3645 }%
3646 {#1}%
3647 }
```

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

```
3648 \newcommand*{\Glsaccessname}[1]{%
3649 \glsnameaccessdisplay
3650 {%
3651 \Glsentryname{#1}%
3652 }%
3653 {#1}%
3654 }
```

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

```
3655 \newcommand*{\GLSaccessname}[1]{%
3656 \glsnameaccessdisplay
3657 {%
3658 \mfirstucMakeUppercase{\glsentryname{#1}}%
3659 }%
3660 {#1}%
3661 }
```

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

```
3662 \newcommand*{\glsaccesstext}[1]{%
3663 \glstextaccessdisplay
3664 {%
3665 \glsentrytext{#1}%
3666 }%
3667 {#1}%
3668 }
```

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

```
3669 \newcommand*{\Glsaccesstext}[1]{%
3670 \glstextaccessdisplay
3671 {%
```

```

3672     \Glstrytext{#1}%
3673   }%
3674   {#1}%
3675 }

```

\GLSaccssextext Display the text value (no link and no check for existence) converted to upper case.

```

3676 \newcommand*{\GLSaccssextext}[1]{%
3677   \glstextaccessdisplay
3678   {%
3679     \mfirstucMakeUppercase{\glstrytext{#1}}%
3680   }%
3681   {#1}%
3682 }

```

glsaccessplural Display the plural value (no link and no check for existence).

```

3683 \newcommand*{\glsaccessplural}[1]{%
3684   \glspluralaccessdisplay
3685   {%
3686     \glstryplural{#1}%
3687   }%
3688   {#1}%
3689 }

```

Glsaccessplural Display the plural value (no link and no check for existence) with the first letter converted to upper case.

```

3690 \newcommand*{\Glsaccessplural}[1]{%
3691   \glspluralaccessdisplay
3692   {%
3693     \Glstryplural{#1}%
3694   }%
3695   {#1}%
3696 }

```

GLSaccessplural Display the plural value (no link and no check for existence) converted to upper case.

```

3697 \newcommand*{\GLSaccessplural}[1]{%
3698   \glspluralaccessdisplay
3699   {%
3700     \mfirstucMakeUppercase{\glstryplural{#1}}%
3701   }%
3702   {#1}%
3703 }

```

\glsaccessfirst Display the first value (no link and no check for existence).

```

3704 \newcommand*{\glsaccessfirst}[1]{%
3705   \glsfirstaccessdisplay
3706   {%
3707     \glstryfirst{#1}%
3708   }%

```

```

3709     {#1}%
3710 }

```

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

```

3711 \newcommand*{\Glsaccessfirst}[1]{%
3712   \glsfirstaccessdisplay
3713   {%
3714     \Glsentryfirst{#1}%
3715   }%
3716   {#1}%
3717 }

```

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

```

3718 \newcommand*{\GLSaccessfirst}[1]{%
3719   \glsfirstaccessdisplay
3720   {%
3721     \mfirstucMakeUppercase{\Glsentryfirst{#1}}%
3722   }%
3723   {#1}%
3724 }

```

`\Glsaccessfirstplural` Display the firstplural value (no link and no check for existence).

```

3725 \newcommand*{\Glsaccessfirstplural}[1]{%
3726   \glsfirstpluralaccessdisplay
3727   {%
3728     \Glsentryfirstplural{#1}%
3729   }%
3730   {#1}%
3731 }

```

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

```

3732 \newcommand*{\Glsaccessfirstplural}[1]{%
3733   \glsfirstpluralaccessdisplay
3734   {%
3735     \Glsentryfirstplural{#1}%
3736   }%
3737   {#1}%
3738 }

```

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

```

3739 \newcommand*{\GLSaccessfirstplural}[1]{%
3740   \glsfirstpluralaccessdisplay
3741   {%
3742     \mfirstucMakeUppercase{\Glsentryfirstplural{#1}}%
3743   }%
3744   {#1}%
3745 }

```

glsaccesssymbol Display the symbol value (no link and no check for existence).

```
3746    \newcommand*{\glsaccesssymbol}[1]{%
3747        \glssymbolaccessdisplay
3748        {%
3749            \glentrysymbol{#1}%
3750        }%
3751        {#1}%
3752    }
```

Glsaccesssymbol Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```
3753    \newcommand*{\Glsaccesssymbol}[1]{%
3754        \glssymbolaccessdisplay
3755        {%
3756            \Glsentrysymbol{#1}%
3757        }%
3758        {#1}%
3759    }
```

GLSaccesssymbol Display the symbol value (no link and no check for existence) converted to upper case.

```
3760    \newcommand*{\GLSaccesssymbol}[1]{%
3761        \glssymbolaccessdisplay
3762        {%
3763            \mfirstucMakeUppercase{\glentrysymbol{#1}}%
3764        }%
3765        {#1}%
3766    }
```

esssymbolplural Display the symbolplural value (no link and no check for existence).

```
3767    \newcommand*{\glsaccesssymbolplural}[1]{%
3768        \glssymbolpluralaccessdisplay
3769        {%
3770            \glentrysymbolplural{#1}%
3771        }%
3772        {#1}%
3773    }
```

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```
3774    \newcommand*{\Glsaccesssymbolplural}[1]{%
3775        \glssymbolpluralaccessdisplay
3776        {%
3777            \Glsentrysymbolplural{#1}%
3778        }%
3779        {#1}%
3780    }
```

esssymbolplural Display the symbolplural value (no link and no check for existence) converted to upper case.

```

3781 \newcommand*{\GLSaccesssymbolplural}[1]{%
3782   \glssymbolpluralaccessdisplay
3783   {%
3784     \mfirstucMakeUppercase{\glsentrysymbolsymbolplural{#1}}%
3785   }%
3786   {#1}%
3787 }

```

`\glsaccessdesc` Display the desc value (no link and no check for existence).

```

3788 \newcommand*{\glsaccessdesc}[1]{%
3789   \glsdescriptionaccessdisplay
3790   {%
3791     \glentrydesc{#1}%
3792   }%
3793   {#1}%
3794 }

```

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

```

3795 \newcommand*{\Glsaccessdesc}[1]{%
3796   \glsdescriptionaccessdisplay
3797   {%
3798     \Glsentrydesc{#1}%
3799   }%
3800   {#1}%
3801 }

```

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

```

3802 \newcommand*{\GLSaccessdesc}[1]{%
3803   \glsdescriptionaccessdisplay
3804   {%
3805     \mfirstucMakeUppercase{\glentrydesc{#1}}%
3806   }%
3807   {#1}%
3808 }

```

`ccessdescplural` Display the descplural value (no link and no check for existence).

```

3809 \newcommand*{\glsaccessdescplural}[1]{%
3810   \glsdescriptionpluralaccessdisplay
3811   {%
3812     \glentrydescplural{#1}%
3813   }%
3814   {#1}%
3815 }

```

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

```

3816 \newcommand*{\Glsaccessdescplural}[1]{%

```

```

3817 \glsdescriptionpluralaccessdisplay
3818 {%
3819     \Glsentrydescplural{#1}%
3820 }%
3821 {#1}%
3822 }

```

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

```

3823 \newcommand*{\Glsaccessplural}[1]{%
3824     \glsdescriptionpluralaccessdisplay
3825     {%
3826         \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
3827     }%
3828     {#1}%
3829 }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

3830 \newcommand*{\glsaccessshort}[1]{%
3831     \glsshortaccessdisplay
3832     {%
3833         \glsentryshort{#1}%
3834     }%
3835     {#1}%
3836 }

```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

3837 \newcommand*{\Glsaccessshort}[1]{%
3838     \glsshortaccessdisplay
3839     {%
3840         \Glsentryshort{#1}%
3841     }%
3842     {#1}%
3843 }

```

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

```

3844 \newcommand*{\GlsAccessshort}[1]{%
3845     \glsshortaccessdisplay
3846     {%
3847         \mfirstucMakeUppercase{\glsentryshort{#1}}%
3848     }%
3849     {#1}%
3850 }

```

`\glsaccessshortpl` Display the short plural form (no link and no check for existence).

```

3851 \newcommand*{\glsaccessshortpl}[1]{%
3852     \glsshortpluralaccessdisplay
3853     {%

```

```

3854     \glentryshortpl{#1}%
3855   }%
3856   {#1}%
3857 }

```

lsaccessshortpl Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

3858 \newcommand*\lsaccessshortpl[1]{%
3859   \glshortpluralaccessdisplay
3860   {%
3861     \glentryshortpl{#1}%
3862   }%
3863   {#1}%
3864 }

```

LSaccessshortpl Display the shortplural value (no link and no check for existence) converted to upper case.

```

3865 \newcommand*\LSaccessshortpl[1]{%
3866   \glshortpluralaccessdisplay
3867   {%
3868     \mfirstucMakeUppercase{\glentryshortpl{#1}}%
3869   }%
3870   {#1}%
3871 }

```

\glaccesslong Display the long form (no link and no check for existence).

```

3872 \newcommand*\glaccesslong[1]{%
3873   \glslongaccessdisplay{\glentrylong{#1}}{#1}%
3874 }

```

\Glsaccesslong Display the long form (no link and no check for existence).

```

3875
3876 \newcommand*\Glsaccesslong[1]{%
3877   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
3878 }

```

\GLSaccesslong Display the long value (no link and no check for existence) converted to upper case.

```

3879 \newcommand*\GLSaccesslong[1]{%
3880   \glslongaccessdisplay
3881   {%
3882     \mfirstucMakeUppercase{\glentrylong{#1}}%
3883   }%
3884   {#1}%
3885 }

```

glsaccesslongpl Display the long plural form (no link and no check for existence).

```

3886 \newcommand*\glsaccesslongpl[1]{%
3887   \glslongpluralaccessdisplay{\glentrylongpl{#1}}{#1}%
3888 }

```

`Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
3889
3890 \newcommand*{\Glsaccesslongpl}[1]{%
3891 \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
3892 }
```

`GLSaccesslongpl` Display the longplural value (no link and no check for existence) converted to upper case.

```
3893 \newcommand*{\GLSaccesslongpl}[1]{%
3894 \glslongpluralaccessdisplay
3895 {%
3896 \mfirstucMakeUppercase{\Glsentrylongpl{#1}}%
3897 }%
3898 {#1}%
3899 }
```

End of if part

```
3900 }
3901 {
```

No accessibility support. Just define these commands to do `\glsentry<xxx>`

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

```
3902 \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.

```
3903 \newcommand*{\GLsaccessname}[1]{\Glsentryname{#1}}
```

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

```
3904 \newcommand*{\GLSAccessname}[1]{%
3905 \protect\mfirstucMakeUppercase{\glsentryname{#1}}}
```

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

```
3906 \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.

```
3907 \newcommand*{\GLsaccesstext}[1]{\Glsentrytext{#1}}
```

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

```
3908 \newcommand*{\GLSAccesstext}[1]{%
3909 \protect\mfirstucMakeUppercase{\glsentrytext{#1}}}
```

`glsaccessplural` Display the plural value (no link and no check for existence).

```
3910 \newcommand*{\glsaccessplural}[1]{\glsentryplural{#1}}
```

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

```
3911 \newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}
```


GLSaccessplural Display the plural value (no link and no check for existence). converted to upper case.

```

3912 \newcommand*{\GLSaccessplural}[1]{%
3913 \protect\mfirstucMakeUppercase{\glentryplural{#1}}}
```

\glaccessfirst Display the first value (no link and no check for existence).

```

3914 \newcommand*{\glaccessfirst}[1]{\glentryfirst{#1}}
```

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

```

3915 \newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}
```

\GLSaccessfirst Display the first value (no link and no check for existence). converted to upper case.

```

3916 \newcommand*{\GLSaccessfirst}[1]{%
3917 \protect\mfirstucMakeUppercase{\glentryfirst{#1}}}
```

cessfirstplural Display the firstplural value (no link and no check for existence).

```

3918 \newcommand*{\glaccessfirstplural}[1]{\glentryfirstplural{#1}}
```

cessfirstplural Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.

```

3919 \newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}
```

cessfirstplural Display the firstplural value (no link and no check for existence). converted to upper case.

```

3920 \newcommand*{\GLSaccessfirstplural}[1]{%
3921 \protect\mfirstucMakeUppercase{\glentryfirstplural{#1}}}
```

glaccesssymbol Display the symbol value (no link and no check for existence).

```

3922 \newcommand*{\glaccesssymbol}[1]{\glentrysymbol{#1}}
```

Glsaccesssymbol Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```

3923 \newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}
```

GLSaccesssymbol Display the symbol value (no link and no check for existence). converted to upper case.

```

3924 \newcommand*{\GLSaccesssymbol}[1]{%
3925 \protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}
```

esssymbolplural Display the symbolplural value (no link and no check for existence).

```

3926 \newcommand*{\glaccesssymbolplural}[1]{\glentrysymbolplural{#1}}
```

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

3927 \newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}
```

esssymbolplural Display the symbolplural value (no link and no check for existence). converted to upper case.

```

3928 \newcommand*{\GLSaccesssymbolplural}[1]{%
3929 \protect\mfirstucMakeUppercase{\glentrysymbolplural{#1}}}
```

`\glsaccessdesc` Display the desc value (no link and no check for existence).
3930 `\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.
3931 `\newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence). converted to upper case.
3932 `\newcommand*{\GLSaccessdesc}[1]{%`
3933 `\protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}`

`ccessdescplural` Display the descplural value (no link and no check for existence).
3934 `\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.
3935 `\newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}`

`ccessdescplural` Display the descplural value (no link and no check for existence). converted to upper case.
3936 `\newcommand*{\GLSaccessdescplural}[1]{%`
3937 `\protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}`

`\glsaccessshort` Display the short form (no link and no check for existence).
3938 `\newcommand*{\glsaccessshort}[1]{\glsentryshort{#1}}`

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).
3939 `\newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}`

`\GLSaccessshort` Display the short value (no link and no check for existence). converted to upper case.
3940 `\newcommand*{\GLSaccessshort}[1]{%`
3941 `\protect\mfirstucMakeUppercase{\glsentryshort{#1}}}`

`lsaccessshortpl` Display the short plural form (no link and no check for existence).
3942 `\newcommand*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}`

`lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).
3943 `\newcommand*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}`

`LSaccessshortpl` Display the shortplural value (no link and no check for existence). converted to upper case.
3944 `\newcommand*{\GLSaccessshortpl}[1]{%`
3945 `\protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}`

`\glsaccesslong` Display the long form (no link and no check for existence).
3946 `\newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}`

`\Glsaccesslong` Display the long form (no link and no check for existence).

```
3947 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

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

```
3948 \newcommand*{\GLSaccesslong}[1]{%
3949 \protect\mfirstucMakeUppercase{\Glsentrylong{#1}}}
```

`glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
3950 \newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}
```

`GLsaccesslongpl` Display the long plural form (no link and no check for existence).

```
3951 \newcommand*{\GLsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

`GLSaccesslongpl` Display the longplural value (no link and no check for existence). converted to upper case.

```
3952 \newcommand*{\GLSaccesslongpl}[1]{%
3953 \protect\mfirstucMakeUppercase{\glsentrylongpl{#1}}}
```

End of else part

```
3954 }
```

1.5 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.

```
3955 \glsaddstoragekey{category}{general}{\glscategory}
```

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.

```
3956 \newcommand{\glsifcategory}[4]{%
3957 \ifglsseldeq{#1}{category}{#2}{#3}{#4}%
3958 }
```

Categories can have attributes.

`categoryattribute` `\glssetcategoryattribute{<category>}{<attribute-label>}{<value>}`

Set (or override if already set) an attribute for the given category.

```
3959 \newcommand*{\glssetcategoryattribute}[3]{%
3960 \csdef{@glsextr@categoryattr@@#1@#2}{#3}%
3961 }
```

`categoryattribute` `\glsgetcategoryattribute{<category>}{<attribute-label>}`

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
3962 \newcommand*{\glsggetcategoryattribute}[2]{%
3963   \csuse{@glsxtr@categoryattr@#1@#2}%
3964 }
```

categoryattribute `\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}`

Tests if the category has the given attribute set.

```
3965 \newcommand*{\glshascategoryattribute}[4]{%
3966   \ifcsvoid{@glsxtr@categoryattr@#1@#2}{#4}{#3}%
3967 }
```

\glsssetattribute `\glsssetattribute{<entry label>}{<attribute-label>}{<value>}`

Short cut where the category label is obtained from the entry information.

```
3968 \newcommand*{\glsssetattribute}[3]{%
3969   \glsssetcategoryattribute{\glscategory{#1}}{#2}{#3}%
3970 }
```

\glsgsetattribute `\glsgsetattribute{<entry label>}{<attribute-label>}`

Short cut where the category label is obtained from the entry information.

```
3971 \newcommand*{\glsgsetattribute}[2]{%
3972   \glsgsetcategoryattribute{\glscategory{#1}}{#2}%
3973 }
```

\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.

```
3974 \newcommand*{\glshasattribute}[4]{%
3975   \ifglentryexists{#1}%
3976   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
3977   {#4}%
3978 }
```

categoryattribute

```
\glsifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}
```

True if category has the attribute with the given value.

```
3979 \newcommand{\glsifcategoryattribute}[5]{%
3980   \ifcsundef{@glstr@categoryattr@#1@#2}%
3981   {#5}%
3982   {\ifcsstring{@glstr@categoryattr@#1@#2}{#3}{#4}{#5}}%
3983 }
```

\glsifattribute

```
\glsifattribute{<entry label>}{<attribute-label>}{<value>}{<true part>}{<false part>}
```

Short cut to determine if the given entry has a category with the given attribute set.

```
3984 \newcommand{\glsifattribute}[5]{%
3985   \ifglentryexists{#1}%
3986   {\glsifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
3987   {#5}%
3988 }
```

Set attributes for the default general category:

```
3989 \glsetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
3990 \glsetcategoryattribute{acronym}{regular}{true}
```

regularcategory

Convenient shortcut to create add the regular attribute.

```
3991 \newcommand*{\glsetregularcategory}[1]{%
3992   \glsetcategoryattribute{#1}{regular}{true}%
3993 }
```

ifregularcategory

```
\glsifregularcategory{<category>}{<true part>}{<false part>}
```

Short cut to determine if a category has the regular attribute explicitly set to true.

```
3994 \newcommand{\glsifregularcategory}[3]{%
3995   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
3996 }
```

ifnotregularcategory

```
\glsifnotregularcategory{<category>}{<true part>}{<false part>}
```

Short cut to determine if a category has the regular attribute explicitly set to false.

```
3997 \newcommand{\glsifnotregularcategory}[3]{%
3998   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
3999 }
```

`\glsifregular` `\glsifregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
4000 \newcommand{\glsifregular}[3]{%
4001   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
4002 }
```

`\glsifnotregular` `\glsifnotregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
4003 \newcommand{\glsifnotregular}[3]{%
4004   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
4005 }
```

`\glsforeachincategory` `\glsforeachincategory[<glossary labels>]{<category-label>}{<glossary-cs>}{<label-cs>}{<body>}`

Iterates through all entries in all the glossaries (or just those listed in *<glossary labels>*) and does *<body>* if the category matches *<category-label>*. The control sequences *<glossary-cs>* and *<label-cs>* may be used in *<body>* to access the glossary label and entry label for the current iteration.

```
4006 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
4007   \forallglossaries[#1]{#3}%
4008   {%
4009     \forglsentries[#3]{#4}%
4010     {%
4011       \glsifcategory{#4}{#2}{#5}{}%
4012     }%
4013   }%
4014 }
```

`\glsforeachwithattribute` `\glsforeachwithattribute[<glossary labels>]{<attribute-label>}{<attribute-value>}{<glossary-cs>}{<label-cs>}{<body>}`

Iterates through all entries in all the glossaries (or just those listed in *⟨glossary labels⟩*) and does *⟨body⟩* if the category attribute *⟨attribute-label⟩* matches *⟨attribute-value⟩*. The control sequences *⟨glossary-cs⟩* and *⟨label-cs⟩* may be used in *⟨body⟩* to access the glossary label and entry label for the current iteration.

```

4015 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
4016   \forallglossaries[#1]{#4}%
4017   {%
4018     \forglsentries[#4]{#5}%
4019     {%
4020       \glsifattribute{#5}{#2}{#3}{#6}{}%
4021     }%
4022   }%
4023 }

```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to index and add `\glsxtrpostdescription`.

```

4024 \ifdef\newterm
4025 {%

```

`\newterm`

```

4026   \renewcommand*{\newterm}[2][ ]{%
4027     \newglossaryentry{#2}%
4028     {type={index},category=index,name={#2},%
4029      description={\glsxtrpostdescription\nopostdesc},#1}%
4030   }

```

Indexed terms are regular by default.

```

4031   \glssetcategoryattribute{index}{regular}{true}

```

`trpostdescindex`

```

4032   \newcommand*{\glsxtrpostdescindex}{}
4033 }
4034 {}

```

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`.

```

4035 \ifdef\printsymbols
4036 {%

```

`glsxtrnewsymbol` Unlike `\newterm`, this has a separate argument for the label (since the symbol will likely contain commands).

```

4037   \newcommand*{\glsxtrnewsymbol}[3][ ]{%
4038     \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
4039   }

```

Symbols are regular by default.

```
4040 \glssetcategoryattribute{symbol}{regular}{true}
```

rpostdescsymbol

```
4041 \newcommand*{\glsxtrpostdescsymbol}{}

4042 }
4043 {}
```

Similar for the numbers option.

```
4044 \ifdef\printnumbers
```

```
4045 {%
```

glsxtrnewnumber

```
4046 \ifdef\printnumbers
```

```
4047 \newcommand*{\glsxtrnewnumber}[3] [] {%
```

```
4048 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
```

```
4049 }
```

Numbers are regular by default.

```
4050 \glssetcategoryattribute{number}{regular}{true}
```

rpostdescnumber

```
4051 \newcommand*{\glsxtrpostdescnumber}{}

4052 }
4053 {}
```

setcategory Set the category for all listed labels. The first argument is the list of entry labels and the second argument is the category label.

```
4054 \newcommand*{\glsxtrsetcategory}[2] {%
```

```
4055 \@for\@glsxtr@label:=#1\do
```

```
4056 {%
```

```
4057 \glsfieldxddef{\@glsxtr@label}{category}{#2}%
```

```
4058 }%
```

```
4059 }
```

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.

```
4060 \newcommand*{\glsxtrsetcategoryforall}[2] {%
```

```
4061 \forallglossaries[#1]{\@glsxtr@type}{%
```

```
4062 \forlentries[\@glsxtr@type]{\@glsxtr@label}%
```

```
4063 {%
```

```
4064 \glsfieldxddef{\@glsxtr@label}{category}{#2}%
```

```
4065 }%
```

```
4066 }%
```

```
4067 }
```


trfieldtitlecase

```
\glstrfieldtitlecase{<label>}{<field>}
```

Apply title casing to the contents of the given field.

```
4068 \newcommand*{\glstrfieldtitlecase}[2]{%
4069   \expandafter\glstrfieldtitlecasesecs\expandafter
4070   {\csname glo@\glsetoklabel{#1}@\#2\endcsname}%
4071 }
```

fieldtitlecasesecs

The command used by `\glstrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```
4072 \newcommand*{\glstrfieldtitlecasesecs}[1]{\xcapitalisewords{#1}}
```

Provide a convenient way to modify glossary styles without having to define a new style just to convert the first letter of fields to upper case.

\glossentrydesc

If the `glossdesc` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```
4073 \@ifpackageloaded{glossaries-accsupp}
4074 {
4075   \renewcommand*{\glossentrydesc}[1]{%
4076     \glsoifexistsorwarn{#1}%
4077     {%
4078       \glsetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the `glossdescfont` attribute to determine the font applied.

```
4079     \glshasattribute{#1}{glossdescfont}%
4080     {%
4081       \edef\@glstr@attrval{\glsetattribute{#1}{glossdescfont}}%
4082       \ifcsdef{\@glstr@attrval}%
4083       {%
4084         \letcs{\@glstr@glossdescfont}{\@glstr@attrval}%
4085       }%
4086       {%
4087         \GlossariesExtraWarning{Unknown control sequence name
4088           ‘\@glstr@attrval’ supplied in glossdescfont attribute
4089           for entry ‘#1’. Ignoring}%
4090         \let\@glstr@glossdescfont\@firstofone
4091       }%
4092     }%
4093     {\let\@glstr@glossdescfont\@firstofone}%
4094     \gl@ifattribute{#1}{glossdesc}{firstuc}%
4095     {%
4096       \@glstr@glossdescfont{\Glsaccessdesc{#1}}%
4097     }%
4098     {%
4099       \gl@ifattribute{#1}{glossdesc}{title}%
4100     }%
```

```

4101         \@glsxtr@do@titlecaps@warn
4102         \glsdescriptionaccessdisplay
4103         {%
4104         \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}}%
4105         }%
4106         {#1}%
4107     }%
4108     {%
4109     \@glsxtr@glossdescfont{\glsaccessdesc{#1}}}%
4110     }%
4111 }%
4112 }%
4113 }
4114 }
4115 {
4116 \renewcommand*{\glossentrydesc}[1]{%
4117 \glsdoifexistsorwarn{#1}%
4118 {%
4119 \glssetabbrvfmt{\glscategory{#1}}%
4120 \glsattribute{#1}{glossdescfont}%
4121 {%
4122 \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
4123 \ifcsdef{\@glsxtr@attrval}%
4124 {%
4125 \letcs{\@glsxtr@glossdescfont}{\@glsxtr@attrval}%
4126 }%
4127 {%
4128 \GlossariesExtraWarning{Unknown control sequence name
4129 '\@glsxtr@attrval' supplied in glossdescfont attribute
4130 for entry '#1'. Ignoring}%
4131 \let\@glsxtr@glossdescfont\@firstofone
4132 }%
4133 }%
4134 {\let\@glsxtr@glossdescfont\@firstofone}%
4135 \glsattribute{#1}{glossdesc}{firstuc}%
4136 {%
4137 \@glsxtr@glossdescfont{\Glsentrydesc{#1}}}%
4138 }%
4139 {%
4140 \glsattribute{#1}{glossdesc}{title}%
4141 {%
4142 \@glsxtr@do@titlecaps@warn
4143 \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}}%
4144 }%
4145 {%
4146 \@glsxtr@glossdescfont{\Glsentrydesc{#1}}}%
4147 }%
4148 }%
4149 }%

```

```

4150 }
4151 }

```

`\glossentryname` If the `glossname` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

4152 \@ifpackageloaded{glossaries-accsupp}
4153 {
4154   \renewcommand*{\glossentryname}[1]{%
4155     \@glsdoifexistsorwarn{#1}%
4156     {%
4157       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```

4158       \glsattribute{#1}{glossnamefont}%
4159       {%
4160         \edef\@glsextr@attrval{\glsglobalattribute{#1}{glossnamefont}}%
4161         \ifcsdef{\@glsextr@attrval}%
4162         {%
4163           \letcs{\@glsextr@glossnamefont}{\@glsextr@attrval}%
4164         }%
4165         {%
4166           \GlossariesExtraWarning{Unknown control sequence name
4167             ‘\@glsextr@attrval’ supplied in glossnamefont attribute
4168             for entry ‘#1’. Reverting to default \string\glsnamefont}%
4169           \let\@glsextr@glossnamefont\glsnamefont
4170         }%
4171       }%
4172       {\let\@glsextr@glossnamefont\glsnamefont}%
4173       \glsifattribute{#1}{glossname}{firstuc}%
4174       {%
4175         \glsnameaccessdisplay
4176         {%
4177           \@glsextr@glossnamefont{\Glsentryname{#1}}%
4178         }%
4179         {#1}%
4180       }%
4181       {%
4182         \glsifattribute{#1}{glossname}{title}%
4183         {%
4184           \@glsextr@do@titlecaps@warn
4185           \glsnameaccessdisplay
4186           {%
4187             \@glsextr@glossnamefont{\glsextr@fieldtitlecase{#1}{name}}%
4188           }%
4189           {#1}%
4190         }%
4191         {%
4192           \glsifattribute{#1}{glossname}{uc}%
4193           {%
4194             \glsnameaccessdisplay

```

```

4195         {%
Hide the label from the upper-casing command.
4196         \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
4197         \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
4198         }%
4199         {#1}%
4200     }%
4201     {%
4202         \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
4203         \glsnameaccessdisplay
4204         {%
4205             \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
4206             }%
4207             {#1}%
4208         }%
4209     }%
4210 }%

Do post-name hook:
4211     \glsxtrpostnamehook{#1}%
4212 }%
4213 }
4214 }
4215 {
4216     \renewcommand*{\glossentryname}[1]{%
4217         \@glsdoifexistsorwarn{#1}%
4218         {%
4219             \glssetabbrvfmt{\glscategory{#1}}%
4220             \glsattribute{#1}{glossnamefont}%
4221             {%
4222                 \edef\@glsxtr@attrval{\glsattribute{#1}{glossnamefont}}%
4223                 \ifcsdef{\@glsxtr@attrval}%
4224                 {%
4225                     \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
4226                 }%
4227                 {%
4228                     \GlossariesExtraWarning{Unknown control sequence name
4229                     '\@glsxtr@attrval' supplied in glossnamefont attribute
4230                     for entry '#1'. Reverting to default \string\glsnamefont}%
4231                     \let\@glsxtr@glossnamefont\glsnamefont
4232                 }%
4233             }%
4234             {\let\@glsxtr@glossnamefont\glsnamefont}%
4235             \glsifattribute{#1}{glossname}{firstuc}%
4236             {%
4237                 \@glsxtr@glossnamefont{\Glsentryname{#1}}%
4238             }%
4239             {%
4240                 \glsifattribute{#1}{glossname}{title}%

```

```

4241      {%
4242      \@glstr@do@titlecaps@warn
4243      \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}}%
4244      }%
4245      {%
4246      \gl@ifattribute{#1}{glossname}{uc}%
4247      {%

```

Hide the label from the upper-casing command.

```

4248      \letcs{\glo@name}{glo@glsetoklabel{#1}@name}%
4249      \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
4250      }%
4251      {%

```

This little trick is used by glossaries to allow the user to redefine \glsnamefont to use \makefirstuc. Support it even though they can now use the firstuc attribute.

```

4252      \letcs{\glo@name}{glo@glsetoklabel{#1}@name}%
4253      \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
4254      }%
4255      }%
4256      }%

```

Do post-name hook.

```

4257      \glstrpostnamehook{#1}%
4258      }%
4259  }
4260 }

```

\Glossentryname Redefine to set the abbreviation format and accessibility support.

```

4261 \@ifpackageloaded{glossaries-accsupp}
4262 {
4263   \renewcommand*{\Glossentryname}[1]{%
4264     \@glsetoklabel{#1}%
4265     {%
4266       \glsetabbrvfmt{\glscategory{#1}}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

4267     \glshasattribute{#1}{glossnamefont}%
4268     {%
4269       \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
4270       \ifcsdef{\@glstr@attrval}%
4271       {%
4272         \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
4273         }%
4274         {%
4275           \GlossariesExtraWarning{Unknown control sequence name
4276             '\@glstr@attrval' supplied in glossnamefont attribute
4277             for entry '#1'. Reverting to default \string\glsnamefont}%
4278           \let\@glstr@glossnamefont\glsnamefont
4279         }%
4280         }%

```

```

4281      {\let\@glxtr@glossnamefont\glsnamefont}%
4282      \glsnameaccessdisplay
4283      {%
4284      \@glxtr@glossnamefont{\Glsentryname{#1}}%
4285      }%
4286      {#1}%

```

Do post-name hook:

```

4287      \glxtrpostnamehook{#1}%
4288      }%
4289  }
4290 }
4291 {
4292  \renewcommand*{\Glossentryname}[1]{%
4293  \@glsdoifexistsorwarn{#1}%
4294  {%
4295  \glssetabbrvfmt{\glscategory{#1}}%
4296  \glsattribute{#1}{glossnamefont}%
4297  {%
4298  \edef\@glxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
4299  \ifcsdef{\@glxtr@attrval}%
4300  {%
4301  \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
4302  }%
4303  {%
4304  \GlossariesExtraWarning{Unknown control sequence name
4305  '\@glxtr@attrval' supplied in glossnamefont attribute
4306  for entry '#1'. Reverting to default \string\glsnamefont}%
4307  \let\@glxtr@glossnamefont\glsnamefont
4308  }%
4309  }%
4310  {\let\@glxtr@glossnamefont\glsnamefont}%
4311  \@glxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

4312      \glxtrpostnamehook{#1}%
4313      }%
4314  }
4315 }

```

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.

`\glxtrpostnamehook` Hook to append stuff after the name is displayed in the glossary. The argument is the entry's label.

```

4316 \newcommand*{\glxtrpostnamehook}[1]{%
4317  \def\@glsnumberformat{glsnumberformat}%
4318  \glxtrdoautoindexname{#1}{indexname}%

```

Allow categories to hook in here.

```

4319 \csuse{glxstrpostname\glscategory{\glscurrententrylabel}}%
4320 }

```

`format@override` Determines if the format key should override the indexing attribute value.

```

4321 \newif\if@glxstr@format@override
4322 \@glxstr@format@overridefalse

```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

4323 \@ifpackageloaded{hyperref}
4324 {

```

If `hyperref`'s `hyperindex` option is on, then `hyperref` will automatically add `\hyperpage`, so don't add it.

```

4325 \ifHy@hyperindex
4326 \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
4327 \@glxstr@format@override true
4328 \appto\theindex{\let\glshypernumber\@firstofone}}%
4329 }
4330 \else
4331 \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
4332 \@glxstr@format@override true
4333 \appto\theindex{\let\glshypernumber\hyperpage}}%
4334 }
4335 \fi
4336 }
4337 {
4338 \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
4339 \@glxstr@format@override true
4340 }
4341 }
4342 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

`doautoindexname`

```

4343 \newcommand*{\glxstrdoautoindexname}[2]{%
4344 \glshasattribute{#1}{#2}}%
4345 {%

```

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.

```

4346 \@glxstr@autoindex@setname{#1}}%

```

If the attribute value is simply “true” don't add an `encap`, otherwise use the value as the `encap`.

```

4347 \protected@edef\@glxstr@attrval{\glsggetattribute{#1}{#2}}%
4348 \if@glxstr@format@override
4349 \ifdefstring{\@glsnnumberformat}{glsnnumberformat}{}%
4350 {\let\@glxstr@attrval\@glsnnumberformat}%
4351 \fi

```

```

4352 \ifdefstring{\@glsxtr@attrval}{true}%
4353 {}%
4354 {\eappto\@glo@name{\@glsxtr@autoindex@encap\@glsxtr@attrval}}%
4355 \expandafter\index\expandafter{\@glo@name}%
4356 }%
4357 {}%
4358 }

```

toindex@setname Assign \@glo@name for use with indexname attribute.

```

4359 \newcommand*{\@glsxtr@autoindex@setname}[1]{%
4360 \def\@glo@name{\string\glsentryname{#1}}%
4361 \glsletentryfield{\@glo@sort}{#1}{sort}%
4362 \@gls@checkmkidxchars\@glo@sort
4363 \@glsxtr@autoindex@doextra@esc\@glo@sort
4364 \epreto\@glo@name{\@glo@sort\@glsxtr@autoindex@at}%
4365 }

```

dex@doextra@esc

```

4366 \newcommand*{\@glsxtr@autoindex@doextra@esc}[1]{%
    Escape the escape character unless it has already been escaped.
4367 \ifx\@glsxtr@autoindex@esc\@gls@quotechar
4368 \else
4369 \def\@gls@checkedmkidx{}%
4370 \edef\@glsxtr@checkspch{%
4371 \noexpand\@glsxtr@autoindex@escquote\expandonce{#1}%
4372 \noexpand\@empty\@glsxtr@autoindex@esc\noexpand\@nnil
4373 \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4374 @@glsxtr@checkspch
4375 \let#1\@gls@checkedmkidx\relax
4376 \fi

```

Escape actual character unless it has already been escaped.

```

4377 \ifx\@glsxtr@autoindex@at\@gls@actualchar
4378 \else
4379 \def\@gls@checkedmkidx{}%
4380 \edef\@glsxtr@checkspch{%
4381 \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
4382 \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
4383 \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4384 @@glsxtr@checkspch
4385 \let#1\@gls@checkedmkidx\relax
4386 \fi

```

Escape level character unless it has already been escaped.

```

4387 \ifx\@glsxtr@autoindex@level\@gls@levelchar
4388 \else
4389 \def\@gls@checkedmkidx{}%
4390 \edef\@glsxtr@checkspch{%
4391 \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%

```



```

4392      \noexpand\@empty\@glxtr@autoindex@level\noexpand\@nnil
4393      \@glxtr@autoindex@level\noexpand\@empty\noexpand\@glxtr@endescspch}%
4394      \@glxtr@checkspch
4395      \let#1\@glxtr@checkedmkidx\relax
4396 \fi

```

Escape encap character unless it has already been escaped.

```

4397 \ifx\@glxtr@autoindex@encap\@glxtr@encapchar
4398 \else
4399 \def\@glxtr@checkedmkidx{%
4400 \edef\@glxtr@checkspch{%
4401 \noexpand\@glxtr@autoindex@escencap\expandonce{#1}%
4402 \noexpand\@empty\@glxtr@autoindex@encap\noexpand\@nnil
4403 \@glxtr@autoindex@encap\noexpand\@empty\noexpand\@glxtr@endescspch}%
4404 \@glxtr@checkspch
4405 \let#1\@glxtr@checkedmkidx\relax
4406 \fi
4407 }

```

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`.

```

4408 \newcommand*{\@glxtr@autoindex@at}{}

```

`trSetActualChar` Set the actual character.

```

4409 \newcommand*{\GlsXtrSetActualChar}[1]{%
4410 \gdef\@glxtr@autoindex@at{#1}%
4411 \def\@glxtr@autoindex@escat##1##2##3\@glxtr@endescspch{%
4412 \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escat}{##1}{##2}{##3}%
4413 }%
4414 }
4415 \@onlypreamble\GlsXtrSetActualChar
4416 \makeatother
4417 \GlsXtrSetActualChar{@}
4418 \makeatletter

```

`autoindex@encap` Encap character for use with `\index`.

```

4419 \newcommand*{\@glxtr@autoindex@encap}{}

```

`XtrSetEncapChar` Set the encap character.

```

4420 \newcommand*{\GlsXtrSetEncapChar}[1]{%
4421 \gdef\@glxtr@autoindex@encap{#1}%
4422 \def\@glxtr@autoindex@escencap##1##2##3\@glxtr@endescspch{%
4423 \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escencap}{##1}{##2}{##3}%
4424 }%
4425 }
4426 \GlsXtrSetEncapChar{|}
4427 \@onlypreamble\GlsXtrSetEncapChar

```

autoindex@level Level character for use with \index.

```
4428 \newcommand*{\@glxtr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
4429 \newcommand*{\GlsXtrSetLevelChar}[1]{%
4430   \gdef\@glxtr@autoindex@level{#1}%
4431   \def\@glxtr@autoindex@esclevel##1#1##2#1##3\@glxtr@endescspch{%
4432     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@esclevel}{##1}{##2}{##3}%
4433   }%
4434 }
4435 \GlsXtrSetLevelChar{!}
4436 \@onlypreamble\GlsXtrSetLevelChar
```

r@autoindex@esc Escape character for use with \index.

```
4437 \newcommand*{\@glxtr@autoindex@esc}{"
```

lsXtrSetEscChar Set the escape character.

```
4438 \newcommand*{\GlsXtrSetEscChar}[1]{%
4439   \gdef\@glxtr@autoindex@esc{#1}%
4440   \def\@glxtr@autoindex@escquote##1#1##2#1##3\@glxtr@endescspch{%
4441     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escquote}{##1}{##2}{##3}%
4442   }%
4443 }
4444 \GlsXtrSetEscChar{"}
4445 \@onlypreamble\GlsXtrSetEscChar
```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```
4446 \ifdef\actualchar
4447 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
4448 {}
```

Quote character \quotechar:

```
4449 \ifdef\quotechar
4450 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
4451 {}
```

Level character \levelchar:

```
4452 \ifdef\levelchar
4453 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
4454 {}
```

Encap character \encapchar:

```
4455 \ifdef\encapchar
4456 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
4457 {}
```

leto@endescspch

```
4458 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

toindex@esc@spch

`\@@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}`

```
4459 \newcommand*{\@@glxtr@autoindex@escspch}[5]{%
4460   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
4461   \toks@={#3}%
4462   \ifx\@nnil#3\relax
4463     \def\@@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
4464   \else
4465     \ifx\@nnil#4\relax
4466       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
4467       \def\@@glxtr@checkspch{\@glxtr@gobbleto@endescspch
4468         #4#5\@glxtr@endescspch}%
4469     \else
4470       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
4471         \@glxtr@autoindex@esc#1}%
4472       \def\@@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
4473     \fi
4474   \fi
4475   \@@glxtr@checkspch
4476 }
```

`\Glossentrydesc` Redefine to set the abbreviation format and accessibility support.

```
4477 \renewcommand*{\Glossentrydesc}[1]{%
4478   \glsdoifexistsorwarn{#1}%
4479   {%
4480     \glssetabbrvfmt{\glscategory{#1}}%
4481     \Glsaccessdesc{#1}%
4482   }%
4483 }
```

`\Glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```
4484 \renewcommand*{\Glossentrysymbol}[1]{%
4485   \glsdoifexistsorwarn{#1}%
4486   {%
4487     \glssetabbrvfmt{\glscategory{#1}}%
4488     \Glsaccesssymbol{#1}%
4489   }%
4490 }
```

`\Glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```
4491 \renewcommand*{\Glossentrysymbol}[1]{%
4492   \glsdoifexistsorwarn{#1}%
4493   {%
4494     \glssetabbrvfmt{\glscategory{#1}}%
4495     \Glsaccesssymbol{#1}%
4496   }%
4497 }
```

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.

```
4498 \newcommand*{\GlsXtrEnableInitialTagging}{%
4499   \ifstar\s@glsxtr@enabletagging\@glsxtr@enabletagging
4500 }
4501 \@onlypreamble\GlsXtrEnableInitialTagging
```

`\r@enabletagging` Starred version undefines command.

```
4502 \newcommand*{\s@glsxtr@enabletagging}[2]{%
4503   \undef#2%
4504   \@glsxtr@enabletagging{#1}{#2}%
4505 }
```

`\r@enabletagging` Internal command.

```
4506 \newcommand*{\@glsxtr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
4507   \@for\@glsxtr@cat:=#1\do
4508   {%
4509     \ifdefempty\@glsxtr@cat
4510     {%
4511       {\glsssetcategoryattribute{\@glsxtr@cat}{tagging}{true}}%
4512     }%
4513     \newrobustcmd*#2[1]{##1}%
4514     \def\@glsxtr@taggingcs{#2}%
4515     \renewcommand*\@glsxtr@activate@initialtagging{%
4516       \let#2\@glsxtr@tag
4517     }%
4518     \ifundef\@gls@preglossaryhook
4519     {\GlossariesExtraWarning{Initial tagging requires at least
4520       glossaries.sty v4.19 to work correctly}}%
4521     {%
4522   }
```

Are we using an old version of `mfirstuc` that has a bug in `\capitalisewords`? If so, patch it so we don't have a problem with a combination of tagging and title case.

`\mfu@checkword@do` If this command hasn't been defined, then we have pre v2.02 of `mfirstuc`

```
4523 \ifundef\mfu@checkword@do
4524 {
4525   \newcommand*{\mfu@checkword@do}[1]{%
4526     \ifdefstring{\mfu@checkword@arg}{#1}%
4527     {%
4528       \let\@mfu@domakefirstuc\@firstofone
4529       \listbreak
4530     }%
4531   }
```

```

4531 {}%
4532 }

```

`\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.

```

4533 \ifundef\mfu@checkword
4534 {
4535   \newcommand{\@glstr@do@titlecaps@warn}{%
4536     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
4537       support not available}%

```

One warning should suffice.

```

4538   \let\@glstr@do@titlecaps@warn\relax
4539 }
4540 }
4541 {
4542   \renewcommand*\mfu@checkword[1]{%
4543     \def\mfu@checkword@arg{#1}%
4544     \let\@mfu@domakefirstuc\makefirstuc
4545     \forlistloop\mfu@checkword@do\@mfu@nocaplist
4546   }
4547 }
4548 }
4549 {}% no patch required

```

`@titlecaps@warn` Do warning if title case not supported.

```

4550 \newcommand*\@glstr@do@titlecaps@warn{}

```

`@initialtagging` Used in `\printglossary` but at least v4.19 of `glossaries` required.

```

4551 \newcommand*\@glstr@activate@initialtagging{}

```

`\@glstr@tag` Definition of tagging command when used in glossary.

```

4552 \newrobustcmd*\@glstr@tag[1]{%
4553   \glstrifattribute{glscurrententrylabel}{tagging}{true}%
4554   {\glstrtagfont{#1}}{#1}%
4555 }

```

`\glstrtagfont` Used in the glossary.

```

4556 \newcommand*\glstrtagfont[1]{\underline{#1}}

```

`preglossaryhook` This macro was introduced in `glossaries` version 4.19, so it may not be defined. If it hasn't been defined this feature is unavailable. A check is added for the entry's existence to prevent errors from occurring if the user removes an entry or changes the label, which can interrupt the build process.

```

4557 \ifdef\@glstr@preglossaryhook
4558 {
4559   \renewcommand*\@glstr@preglossaryhook{%
4560     \@glstr@activate@initialtagging

```

Since the glossaries are automatically scoped, `\@glstr@org@postdescription` shouldn't already be defined, but check anyway just as a precautionary measure.

```

4561 \ifundef\@glstr@org@postdescription
4562 {%
4563 \let\@glstr@org@postdescription\glspostdescription
4564 \renewcommand*{\glspostdescription}{%
4565 \ifglentryexists{\glscurrententrylabel}%
4566 {%
4567 \glstrpostdescription
4568 \@glstr@org@postdescription
4569 }%
4570 }%
4571 }%
4572 }%
4573 {}%
```

Enable the options used by `\@glstrp`:

```

4574 \glossxtrsetpopts
4575 }%
4576 }
4577 {}
```

`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.

```

4578 \newcommand*{\glstrpostdescription}{%
4579 \csuse{glstrpostdesc\glscategory{\glscurrententrylabel}}%
4580 }
```

`postdescgeneral`

```

4581 \newcommand*{\glstrpostdescgeneral}{}%
```

`xtrpostdescterm`

```

4582 \newcommand*{\glstrpostdescterm}{}%
```

`postdescacronym`

```

4583 \newcommand*{\glstrpostdescacronym}{}%
```

`escabbreviation`

```

4584 \newcommand*{\glstrpostdescabbreviation}{}%
```

`glspostlinkhook` Redefine the post link hook used by commands like `\gls` to make it easier for categories or attributes to modify this action. Since this hook occurs outside the existence check of commands like `\gls`, this needs to be checked again here. Do nothing if the entry hasn't been defined.

```

4585 \renewcommand*{\glspostlinkhook}{%
4586 \ifglentryexists{\glslabel}{\glstrpostlinkhook}{}%
4587 }
```

trpostlinkhook The entry label should already be stored in \glslabel by \@gls@link.

```
4588 \newcommand*{\glstrpostlinkhook}{%
4589   \glstrdiscardperiod{\glslabel}%
4590   {\glstrpostlinkendsentence}%
4591   {\glstrpostlink}%
4592 }
```

\glstrpostlink

```
4593 \newcommand*{\glstrpostlink}{%
4594   \csuse{glstrpostlink\glscategory{\glslabel}}%
4595 }
```

linkendsentence Done by \glstrpostlinkhook if a full stop is discarded.

```
4596 \newcommand*{\glstrpostlinkendsentence}{%
4597   \ifcsdef{glstrpostlink\glscategory{\glslabel}}
4598   {%
4599     \csuse{glstrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
4600   .\spacefactor\sfcode'\. \relax
4601   }%
4602   {%
```

Assume the full stop was discarded because the entry ends with a period, so adjust the space-factor.

```
4603   \spacefactor\sfcode'\. \relax
4604   }%
4605 }
```

dDescOnFirstUse Provide a command for appending the description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
4606 \newcommand*{\glstrpostlinkAddDescOnFirstUse}{%
4607   \glstrifwasfirstuse{\space(\glsaccessdesc{\glslabel})}{}%
4608 }
```

ymbolOnFirstUse Provide a command for appending the symbol (if defined) in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
4609 \newcommand*{\glstrpostlinkAddSymbolOnFirstUse}{%
4610   \glstrifwasfirstuse
4611   {%
4612     \ifglshassymbol{\glslabel}{\space(\glsaccesssymbol{\glslabel})}{}%
4613   }%
4614   {}%
4615 }
```

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).

```

4616 \newcommand*{\glxtrdiscardperiod}[3]{%
4617   \glxtrifwasfirstuse
4618   {%
4619     \glusifattribute{#1}{retainfirstuseperiod}{true}%
4620     {#3}%
4621     {%
4622       \glusifattribute{#1}{discardperiod}{true}%
4623       {%
4624         \glusifplural
4625         {%
4626           \glusifattribute{#1}{pluraldiscardperiod}{true}%
4627           {\glxtrifperiod{#2}{#3}}%
4628           {#3}%
4629         }%
4630         {%
4631           \glxtrifperiod{#2}{#3}%
4632         }%
4633       }%
4634     {#3}%
4635   }%
4636 }%
4637 {%
4638   \glusifattribute{#1}{discardperiod}{true}%
4639   {%
4640     \glusifplural
4641     {%
4642       \glusifattribute{#1}{pluraldiscardperiod}{true}%
4643       {\glxtrifperiod{#2}{#3}}%
4644       {#3}%
4645     }%
4646     {%
4647       \glxtrifperiod{#2}{#3}%
4648     }%
4649   }%
4650   {#3}%
4651 }%
4652 }

```

\glxtrifperiod Make a convenient user command to check if the next character is a full stop (period). Works like \@ifstar but uses \new@ifnextchar rather than \@ifnextchar

```
4653 \newcommand*{\glxtrifperiod}[1]{\new@ifnextchar.\{\@firstoftwo{#1}}}
```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

glxtr@punclist List of characters identified as punctuation marks. (Be careful of babel shorthands!) This doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```
4654 \newcommand*{\glxtr@punclist}{.,;?!}
```

punctuationmark Add character to punctuation list.

```
4655 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}
```


unctuationmarks Reset the punctuation list.

```
4656 \newcommand*{\glxtrsetpunctuationmarks}[1]{\def\glxtr@punclist{#1}}
```

```
\glxtrifpunc \glxtrifnextpunc{\<true part>}{\<>false part>}
```

Test if this is followed by a punctuation mark. (Adapted from \new@ifnextchar.)

```
4657 \newcommand*{\glxtrifnextpunc}[2]{%
4658   \def\reserved@a{#1}%
4659   \def\reserved@b{#2}%
4660   \futurelet\@glspunc@token\glxtr@ifnextpunc
4661 }
```

glxtr@ifnextpunc

```
4662 \newcommand*{\glxtr@ifnextpunc}{%
4663   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
4664   \reserved@b
4665 }
```

glxtr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
4666 \newcommand*{\glxtr@ifpunctoken}[1]{%
4667   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
4668 }
```

glxtr@ifpunctoken

```
4669 \def\@glxtr@ifpunctoken#1#2{%
4670   \let\reserved@d=#2%
4671   \ifx\reserved@d\@nnil
4672     \let\glxtr@next\@glxtr@notfoundinlist
4673   \else
4674     \ifx#1\reserved@d
4675       \let\glxtr@next\@glxtr@foundinlist
4676     \else
4677       \let\glxtr@next\@glxtr@ifpunctoken
4678     \fi
4679   \fi
4680   \glxtr@next#1%
4681 }
```

glxtr@foundinlist

```
4682 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
4683 \def\@glxtr@notfoundinlist#1{\@secondoftwo}
```

glsxtrdopostpunc

```
\glsxtrdopostpunc{<code>}
```

If this is followed by a punctuation character, do *<code>* after the character otherwise do *<code>* before whatever comes next.

```
4684 \newcommand{\glsxtrdopostpunc}[1]{%
4685   \glsxtrifnextpunc{\@glsxtr@swaptwo{#1}}{#1}%
4686 }
```

@glsxtr@swaptwo

```
4687 \newcommand{\@glsxtr@swaptwo}[2]{#2#1}
```

1.6 Abbreviations

The “acronym” code from glossaries is misnamed as it’s more often used for other forms of abbreviations. This code corrects this inconsistency, but rather than just having synonyms, provide commands for abbreviations that have a similar, but not identical, underlying mechanism to acronyms.

If there’s a style for the given category, apply it.

```
4688 \define@key{glsxtrabbrv}{category}{%
4689   \edef\glscategorylabel{#1}%
4690   \ifcsdef{@glsabbrv@current@#1}%
4691   {%
```

Warning should already have been issued.

```
4692   \let\@glsxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
4693   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
4694   \glsxtr@applyabbrvstyle{\csname @glsabbrv@current@#1\endcsname}%
4695   \let\GlsXtrWarnDeprecatedAbbrStyle\@glsxtr@orgwarndep
4696   }%
4697   {}%
4698 }
```

Save the short plural form. This may be needed before the entry is defined.

```
4699 \define@key{glsxtrabbrv}{shortplural}{%
4700   \def\@gls@shortpl{#1}%
4701 }
```

Similarly for the long plural form.

```
4702 \define@key{glsxtrabbrv}{longplural}{%
4703   \def\@gls@longpl{#1}%
4704 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```
4705 \newtoks\glsshortpltok
```

```

\glslongpltok
4706 \newtoks\glslongpltok

\glsxtr@insertdots Provided in case user wants to automatically insert dots between each letter of the abbrevi-
ation. This should be applied before defining the abbreviation to optimise the document
build. (Otherwise, it would have to be done each time the short form is required, which is an
unnecessary waste of time.) For this to work the short form must be expanded when passed
to \newabbreviation. Note that explicitly using the short or shortplural keys will override
this.
4707 \newcommand*{\@glsxtr@insertdots}[2]{%
4708   \def#1{%
4709     \@glsxtr@insert@dots#1#2\@nnil
4710   }

\glsxtr@insert@dots
4711 \newcommand*{\@glsxtr@insert@dots}[2]{%
4712   \ifx\@nnil#2\relax
4713   \let\@glsxtr@insert@dots@next\@gobble
4714   \else
4715     \ifx\relax#2\relax
4716     \else
4717       \appto#1{#2.}%
4718     \fi
4719     \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
4720   \fi
4721   \@glsxtr@insert@dots@next#1%
4722 }

\newabbreviation Define a new generic abbreviation.
4723 \newcommand*{\newabbreviation}[4] [] {%
4724   \glsxtr@newabbreviation{#1}{#2}{#3}{#4}%
4725 }

\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.)
4726 \newcommand*{\glsxtr@newabbreviation}[4]{%
4727   \glskeylisttok{#1}%
4728   \glslabeltok{#2}%
4729   \glsshorttok{#3}%
4730   \glslongtok{#4}%

  Get the category.
4731   \def\glscategorylabel{abbreviation}%
4732   \glsxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
4733   \setkeys*{glsxtrabbrv}[shortplural,longplural]{#1}%

  Set the default long plural
4734   \def\@gls@longpl{#4\glspluralsuffix}%

```

Has the insertdots attribute been set?

```
4735 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
4736 {%
4737   \@glxtr@insertdots\@gls@short{#3}%
4738   \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
4739   \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
4740   {%
4741     \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
4742       '\abbrvpluralsuffix}%
4743   }%
4744   {%
4745     \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
4746     {%
4747       \let\@gls@shortpl\@gls@short
4748     }%
4749     {%
4750       \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
4751         \abbrvpluralsuffix}%
4752     }%
4753   }%
4754 }%
4755 {%
```

insertdots not true.

```
4756 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
4757 {%
4758   \def\@gls@shortpl{#3'\abbrvpluralsuffix}%
4759 }%
4760 {%
4761   \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
4762   {%
4763     \def\@gls@shortpl{#3}%
4764   }%
4765   {%
4766     \def\@gls@shortpl{#3\abbrvpluralsuffix}%
4767   }%
4768 }%
4769 }%
```

Hook for further customisation if required:

```
4770 \glxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%
```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary.

```
4771 \setkeys*{glxtrabbrev}[category]{#1}%
```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```
4772 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
4773 \expandafter\glslongpltok\expandafter{\@gls@longpl}%
```

Do any extra setup provided by hook:

```

4774 \newabbreviationhook
    Define this entry:
4775 \protected@edef\@do@newglossaryentry{%
4776 \noexpand\newglossaryentry{\the\glslabeltok}%
4777 {%
4778     type=\glstrabbrvtype,%
4779     category=abbreviation,%
4780     short={\the\glsshorttok},%
4781     shortplural={\the\glsshortpltok},%
4782     long={\the\glslongtok},%
4783     longplural={\the\glslongpltok},%
4784     name={\the\glsshorttok},%
4785     \CustomAbbreviationFields,%
4786     \the\glskeylisttok
4787 }%
4788 }%
4789 \@do@newglossaryentry
4790 \GlsXtrPostNewAbbreviation
4791 }

evpresetkeyhook Hook for extra stuff in \newabbreviation
4792 \newcommand*{\glstrnewabbspresetkeyhook}[3]{}

NewAbbreviation Hook used by abbreviation styles.
4793 \newcommand*{\GlsXtrPostNewAbbreviation}{}

bbreviationhook Hook for use with \newabbreviation.
4794 \newcommand*{\newabbreviationhook}{}

reviationFields
4795 \newcommand*{\CustomAbbreviationFields}{}

lsxtrfullformat Full format without case change.
4796 \newcommand*{\glsxtrfullformat}[2]{%
4797 \glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
4798 (\protect\glsfirstabbrvfont{\glsaccessshort{#1}})%
4799 }

lsxtrfullformat Full format with case change.
4800 \newcommand*{\Glsxtrfullformat}[2]{%
4801 \glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
4802 (\protect\glsfirstabbrvfont{\Glsaccessshort{#1}})%
4803 }

xtrfullplformat Plural full format without case change.
4804 \newcommand*{\glsxtrfullplformat}[2]{%
4805 \glsfirstlongfont{\glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
4806 (\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}})%
4807 }

```

`\glxtrfullplformat` Plural full format with case change.

```
4808 \newcommand*{\Glsxtrfullplformat}[2]{%
4809   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glxtrfullsep{#1}%
4810   (\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}})%
4811 }
```

`\glxtrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
4812 \newcommand*{\glxtrfullsep}[1]{\space}
```

In-line formats in case first use isn't compatible with `\glentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glxtrinfullformat` Full format without case change.

```
4813 \newcommand*{\glxtrinfullformat}{\glxtrfullformat}
```

`\glxtrfullformat` Full format with case change.

```
4814 \newcommand*{\Glsxtrinfullformat}{\Glsxtrfullformat}
```

`\glxtrfullplformat` Plural full format without case change.

```
4815 \newcommand*{\glxtrinfullplformat}{\glxtrfullplformat}
```

`\glxtrfullplformat` Plural full format with case change.

```
4816 \newcommand*{\Glsxtrinfullplformat}{\Glsxtrfullplformat}
```

Redefine `\glentryfull` etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the `\glxtrfull` set of commands instead.

`\glentryfull`

```
4817 \renewcommand*{\glentryfull}[1]{\glxtrinfullformat{#1}{}}
```

`\Glsentryfull`

```
4818 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinfullformat{#1}{}}
```

`\glentryfullpl`

```
4819 \renewcommand*{\glentryfullpl}[1]{\glxtrinfullplformat{#1}{}}
```

`\Glsentryfullpl`

```
4820 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinfullplformat{#1}{}}
```

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
4821 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}
```

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.

```
4822 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}
```

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.

```
4823 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}
```

bbvrdefaultfont

```
4824 \newcommand*{\glsabbrvdefaultfont}[1]{#1}
```

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.

```
4825 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}
```

longdefaultfont Default font changing command used for the long form in commands like `\glsxtrlong`.

```
4826 \newcommand*{\glslongdefaultfont}[1]{#1}
```

lsfirstlongfont Font changing command used for the long form on first use or in the full format.

```
4827 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}
```

longdefaultfont

```
4828 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}
```

brvpluralsuffix Default plural suffix. Allow an alternative default suffix for abbreviations.

```
4829 \newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}
```

brvpluralsuffix Default plural suffix.

```
4830 \newcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}
```

`\glsxtrfull` Full form (no case-change).

```
4831 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\@ns@glsxtrfull}
```

```
4832 \newcommand*{\ns@glsxtrfull}[2][\%
```

```
4833 \new@ifnextchar[{\@glsxtr@full{#1}{#2}}{\%
```

```
4834 {\@glsxtr@full{#1}{#2}[]}\%
```

```
4835 }
```

`\@glsxtr@full` Low-level macro:

```
4836 \def\@glsxtr@full#1#2[#3]{\%
```

```
4837 \glsdoifexists{#2}\%
```

```
4838 {\%
```

```
4839 \glssetabbrvfmt{\gls@category{#2}}{\%
```

```
4840 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
4841 \let\gls@ifplural\@secondoftwo
```

```
4842 \let\gls@caps@case\@firstofthree
```

```
4843 \let\gls@insert\@empty
```

```
4844 \def\gls@customtext{\glsxtr@inline@fullformat{#2}{#3}}{\%
```

What should `\glsxtrifwasfirstuse` be set to here? Where the inline and display full forms are the same, this is essentially emulating first use, to it make sense for the postlink hook to pretend it was a first use instance. It makes less sense if the inline and display forms are different. Provide a hook to make it easier to reconfigure.

```
4845 \glsxtrsetupfulldefs
```

```
4846 \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}\%
```

```
4847 }{\%
```

```
4848 \gls@postlinkhook
```

```
4849 }
```

trsetupfulldefs

```
4850 \newcommand*\glxstrsetupfulldefs{%
4851   \let\glxstrifwasfirstuse\@firstoftwo
4852 }
```

\Glsxtrfull Full form (first letter uppercase).

```
4853 \newrobustcmd*\Glsxtrfull{\@gls@hyp@opt\@ns@Glsxtrfull}
4854 \newcommand*\ns@Glsxtrfull[2][\@]{}
4855   \new@ifnextchar[\@Glsxtr@full{#1}{#2}}%
4856           {\@Glsxtr@full{#1}{#2}[]}%
4857 }
```

\@Glsxtr@full Low-level macro:

```
4858 \def\@Glsxtr@full#1#2[#3]{%
4859   \glsdoifexists{#2}%
4860   {%
4861     \glssetabbrvfmt{\glscategory{#2}}%
4862     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4863     \let\glsifplural\@secondoftwo
4864     \let\glscapscase\@secondofthree
4865     \let\glsinsert\@empty
4866     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
4867     \glxstrsetupfulldefs
4868     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4869   }%
4870   \glspostlinkhook
4871 }
```

\GLSxtrfull Full form (all uppercase).

```
4872 \newrobustcmd*\GLSxtrfull{\@gls@hyp@opt\@ns@GLSxtrfull}
4873 \newcommand*\ns@GLSxtrfull[2][\@]{}
4874   \new@ifnextchar[\@GLSxtr@full{#1}{#2}}%
4875           {\@GLSxtr@full{#1}{#2}[]}%
4876 }
```

\@GLSxtr@full Low-level macro:

```
4877 \def\@GLSxtr@full#1#2[#3]{%
4878   \glsdoifexists{#2}%
4879   {%
4880     \glssetabbrvfmt{\glscategory{#2}}%
4881     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4882     \let\glsifplural\@secondoftwo
4883     \let\glscapscase\@thirdofthree
4884     \let\glsinsert\@empty
4885     \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
4886     \glxstrsetupfulldefs
4887     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4888   }%
4889   \glspostlinkhook
```


4890 }

`\glxstrfullpl` Plural full form (no case-change).

```
4891 \newrobustcmd*{\glxstrfullpl}{\@gls@hyp@opt\@ns@glxstrfullpl}
4892 \newcommand*\ns@glxstrfullpl[2][{}]{%
4893   \new@ifnextchar[{\@glxstr@fullpl{#1}{#2}}{%
4894     {\@glxstr@fullpl{#1}{#2}}}%
4895 }
```

`\@glxstr@fullpl` Low-level macro:

```
4896 \def\@glxstr@fullpl#1#2[#3]{%
4897   \glsdoifexists{#2}%
4898   {%
4899     \glssetabbrvfmt{\glscategory{#2}}%
4900     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4901     \let\glsifplural\@firstoftwo
4902     \let\glscapscase\@firstofthree
4903     \let\glsinsert\@empty
4904     \def\glscustomtext{\glxstrinlinefullplformat{#2}{#3}}%
4905     \glxstrsetupfulldefs
4906     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
4907   }%
4908   \glspostlinkhook
4909 }
```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```
4910 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
4911 \newcommand*\ns@Glsxtrfullpl[2][{}]{%
4912   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}{%
4913     {\@Glsxtr@fullpl{#1}{#2}}}%
4914 }
```

`\@Glsxtr@fullpl` Low-level macro:

```
4915 \def\@Glsxtr@fullpl#1#2[#3]{%
4916   \glsdoifexists{#2}%
4917   {%
4918     \glssetabbrvfmt{\glscategory{#2}}%
4919     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4920     \let\glsifplural\@firstoftwo
4921     \let\glscapscase\@secondofthree
4922     \let\glsinsert\@empty
4923     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
4924     \glxstrsetupfulldefs
4925     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
4926   }%
4927   \glspostlinkhook
4928 }
```

`\GLSxtrfullpl` Plural full form (all upper case).

```

4929 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\ns@GLSxtrfullpl}
4930 \newcommand*\ns@GLSxtrfullpl[2][]{%
4931   \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}]{%
4932     {\@GLSxtr@fullpl{#1}{#2}[]}%
4933 }

```

\@GLSxtr@fullpl Low-level macro:

```

4934 \def\@GLSxtr@fullpl#1#2[#3]{%
4935   \glsdoifexists{#2}%
4936   {%
4937     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4938     \let\glsifplural\@firstoftwo
4939     \let\glscapscase\@thirdofthree
4940     \let\glsinsert\@empty
4941     \def\glscustomtext{%
4942       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
4943     \glsxtrsetupfulldefs
4944     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4945   }%
4946   \glspostlinkhook
4947 }

```

The short and long forms work in a similar way to acronyms.

\glsxtrshort

```

4948 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4949 \newcommand*{\ns@glsxtrshort}[2][]{%
4950   \new@ifnextchar[{\@glsxtrshort{#1}{#2}}{\@glsxtrshort{#1}{#2}[]}%
4951 }

```

Read in the final optional argument:

```

4952 \def\@glsxtrshort#1#2[#3]{%
4953   \glsdoifexists{#2}%
4954   {%

```

Need to make sure \glsabbrvfont is set correctly.

```

4955     \glssetabbrvfmt{\glscategory{#2}}%
4956     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4957     \let\glsxtrifwasfirstuse\@secondoftwo
4958     \let\glsifplural\@secondoftwo
4959     \let\glscapscase\@firstofthree
4960     \let\glsinsert\@empty
4961     \def\glscustomtext{%
4962       \glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
4963       \ifglsxtrininsertinside\else#3\fi
4964     }%
4965     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4966   }%
4967   \glspostlinkhook

```

4968 }

\Glsxtrshort

4969 \newrobustcmd*{\Glsxtrshort}{\@gls@hyp@opt\@ns@Glsxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

4970 \newcommand*{\ns@Glsxtrshort}[2] [] {%

4971 \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} [] }%

4972 }

Read in the final optional argument:

4973 \def\@Glsxtrshort#1#2[#3] {%

4974 \glsdoifexists{#2}%

4975 {%

4976 \glssetabbrvfmt{\glscategory{#2}}%

4977 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

4978 \let\glsxtrifwasfirstuse\@secondoftwo

4979 \let\glsifplural\@secondoftwo

4980 \let\glsapscase\@secondofthree

4981 \let\glsinsert\@empty

4982 \def\glscustomtext{%

4983 \glsabbrvfont{\Glsaccessshort{#2}}\ifglsxtrininsertinside#3\fi}%

4984 \ifglsxtrininsertinside\else#3\fi

4985 }%

4986 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%

4987 }%

4988 \glspostlinkhook

4989 }

\GLSxtrshort

4990 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

4991 \newcommand*{\ns@GLSxtrshort}[2] [] {%

4992 \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} [] }%

4993 }

Read in the final optional argument:

4994 \def\@GLSxtrshort#1#2[#3] {%

4995 \glsdoifexists{#2}%

4996 {%

4997 \glssetabbrvfmt{\glscategory{#2}}%

4998 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

4999 \let\glsxtrifwasfirstuse\@secondoftwo

5000 \let\glsifplural\@secondoftwo

5001 \let\glsapscase\@thirdofthree

5002 \let\glsinsert\@empty

5003 \def\glscustomtext{%

5004 \mfirstucMakeUppercase

5005 {\glsabbrvfont{\Glsaccessshort{#2}}\ifglsxtrininsertinside#3\fi}%

5006 \ifglsxtrininsertinside\else#3\fi

```

5007     }%
5008     }%
5009     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5010     }%
5011     \glspostlinkhook
5012 }

```

`\glsxtrlong`

```

5013 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
5014 \newcommand*{\ns@glsxtrlong}[2][{}]{%
5015   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2}[]}%
5016 }

    Read in the final optional argument:
5017 \def\@glsxtrlong#1#2[#3]{%
5018   \glsdoifexists{#2}%
5019   {%
5020     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5021     \let\glsxtrifwasfirstuse\@secondoftwo
5022     \let\glsifplural\@secondoftwo
5023     \let\glscapscase\@firstofthree
5024     \let\glsinsert\@empty
5025     \def\glscustomtext{%
5026       \glslongfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
5027       \ifglsxtrininsertinside\else#3\fi
5028     }%
5029     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5030     }%
5031     \glspostlinkhook
5032 }

```

`\Glsxtrlong`

```

5033 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
5034 \newcommand*{\ns@Glsxtrlong}[2][{}]{%
5035   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2}[]}%
5036 }

    Read in the final optional argument:
5037 \def\@Glsxtrlong#1#2[#3]{%
5038   \glsdoifexists{#2}%
5039   {%
5040     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5041     \let\glsxtrifwasfirstuse\@secondoftwo
5042     \let\glsifplural\@secondoftwo
5043     \let\glscapscase\@secondofthree
5044     \let\glsinsert\@empty
5045     \def\glscustomtext{%

```

```

5046      \glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
5047      \ifglxtrinsertinside\else#3\fi
5048  }%
5049  \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5050  }%
5051  \glspostlinkhook
5052 }

```

\GLSxtrlong

```

5053 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}

  Define the un-starred form. Need to determine if there is a final optional argument
5054 \newcommand*{\ns@GLSxtrlong}[2] [] {%
5055   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
5056 }

  Read in the final optional argument:
5057 \def\@GLSxtrlong#1#2[#3] {%
5058   \glsdoifexists{#2}%
5059   {%
5060     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5061     \let\glxtrifwasfirstuse\@secondoftwo
5062     \let\glsifplural\@secondoftwo
5063     \let\glscapscase\@thirdofthree
5064     \let\glsinsert\@empty
5065     \def\glscustomtext{%
5066       \mfirstucMakeUppercase
5067       {\glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
5068       \ifglxtrinsertinside\else#3\fi
5069     }%
5070   }%
5071   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5072   }%
5073   \glspostlinkhook
5074 }

```

Plural short forms:

\glxtrshortpl

```

5075 \newrobustcmd*{\glxtrshortpl}{\@gls@hyp@opt\ns@glxtrshortpl}

  Define the un-starred form. Need to determine if there is a final optional argument
5076 \newcommand*{\ns@glxtrshortpl}[2] [] {%
5077   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2} []}%
5078 }

  Read in the final optional argument:
5079 \def\@glxtrshortpl#1#2[#3] {%
5080   \glsdoifexists{#2}%
5081   {%
5082     \glssetabbrvfmt{\glscategory{#2}}%

```

```

5083 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5084 \let\glxtrifwasfirstuse\@secondoftwo
5085 \let\gl@sifplural\@firstoftwo
5086 \let\glscapscase\@firstofthree
5087 \let\gl$insert\@empty
5088 \def\glscustomtext{%
5089 \glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
5090 \ifglxtrininsertinside\else#3\fi
5091 }%
5092 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5093 }%
5094 \glspostlinkhook
5095 }

```

\Glsxtrshortpl

```

5096 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\ns@Glsxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
5097 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
5098 \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
5099 }

    Read in the final optional argument:
5100 \def\@Glsxtrshortpl#1#2[#3] {%
5101 \glsoifexists{#2}%
5102 {%
5103 \glsetabbrvfmt{\glscategory{#2}}%
5104 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5105 \let\glxtrifwasfirstuse\@secondoftwo
5106 \let\gl@sifplural\@firstoftwo
5107 \let\glscapscase\@secondofthree
5108 \let\gl$insert\@empty
5109 \def\glscustomtext{%
5110 \glabbrvfont{\Glsaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
5111 \ifglxtrininsertinside\else#3\fi
5112 }%
5113 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5114 }%
5115 \glspostlinkhook
5116 }

```

\GLSxtrshortpl

```

5117 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
5118 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
5119 \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
5120 }

    Read in the final optional argument:
5121 \def\@GLSxtrshortpl#1#2[#3] {%

```

```

5122 \glsdoifexists{#2}%
5123 {%
5124   \glssetabbrvfmt{\glscategory{#2}}%
5125   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5126   \let\glsxtrifwasfirstuse\@secondoftwo
5127   \let\glsifplural\@firstoftwo
5128   \let\gls caps case\@thirdofthree
5129   \let\glsinsert\@empty
5130   \def\gls custom text{%
5131     \mfirstucMakeUppercase
5132     {\glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
5133     \ifglsxtrininsertinside\else#3\fi
5134   }%
5135 }%
5136 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5137 }%
5138 \gls post link hook
5139 }

```

Plural long forms:

\glsxtrlongpl

```

5140 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\@ns@glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

5141 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
5142   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
5143 }

```

Read in the final optional argument:

```

5144 \def\@glsxtrlongpl#1#2[#3]{%
5145   \glsdoifexists{#2}%
5146   {%
5147     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5148     \let\glsxtrifwasfirstuse\@secondoftwo
5149     \let\glsifplural\@firstoftwo
5150     \let\gls caps case\@firstofthree
5151     \let\glsinsert\@empty
5152     \def\gls custom text{%
5153       \gls long font{\glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
5154       \ifglsxtrininsertinside\else#3\fi
5155     }%
5156     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5157   }%
5158   \gls post link hook
5159 }

```

\Glsxtrlongpl

```

5160 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```
5161 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
5162   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
5163 }
```

Read in the final optional argument:

```
5164 \def\@Glsxtrlongpl#1#2[#3]{%
5165   \glsdoifexists{#2}%
5166   {%
5167     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5168     \let\glsxtrifwasfirstuse\@secondoftwo
5169     \let\glsifplural\@firstoftwo
5170     \let\glscapscase\@secondofthree
5171     \let\glsinsert\@empty
5172     \def\glscustomtext{%
5173       \glslongfont{\Glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
5174       \ifglsxtrininsertinside\else#3\fi
5175     }%
5176     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5177   }%
5178   \glspostlinkhook
5179 }
```

\GLSxtrlongpl

```
5180 \newrobustcmd*{\GLSxtrlongpl}{\@gls@hyp@opt\ns@GLSxtrlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
5181 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
5182   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
5183 }
```

Read in the final optional argument:

```
5184 \def\@GLSxtrlongpl#1#2[#3]{%
5185   \glsdoifexists{#2}%
5186   {%
5187     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5188     \let\glsxtrifwasfirstuse\@secondoftwo
5189     \let\glsifplural\@firstoftwo
5190     \let\glscapscase\@thirdofthree
5191     \let\glsinsert\@empty
5192     \def\glscustomtext{%
5193       \mfirstucMakeUppercase
5194       {\glslongfont{\Glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
5195       \ifglsxtrininsertinside\else#3\fi
5196     }%
5197   }%
5198   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5199 }%
5200 \glspostlinkhook
5201 }
```


`\glssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```
5202 \newcommand*{\glssetabbrvfmt}[1]{%
5203   \ifcsdef{@glsabbrv@current@#1}%
5204     {\glsxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
5205     {\glsxtr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
5206 }
```

`sxtrngenabbrvfmt` Similar to `\glsngenacfmt`, but for abbreviations.

```
5207 \newcommand*{\glsxtrngenabbrvfmt}{%
5208   \ifdefempty\glscustomtext
5209     {%
5210       \ifglsused\glslabel
5211         {%
```

Subsequent use:

```
5212       \glsifplural
5213       {%
```

Subsequent plural form:

```
5214       \glscapscase
5215       {%
```

Subsequent plural form, don't adjust case:

```
5216       \glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert
5217       }%
5218       {%
```

Subsequent plural form, make first letter upper case:

```
5219       \glsabbrvfont{\Glsaccessshortpl{\glslabel}}\glsinsert
5220       }%
5221       {%
```

Subsequent plural form, all caps:

```
5222       \mfirstucMakeUppercase
5223       {\glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert}%
5224       }%
5225       }%
5226       {%
```

Subsequent singular form

```
5227       \glscapscase
5228       {%
```

Subsequent singular form, don't adjust case:

```
5229       \glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert
5230       }%
5231       {%
```

Subsequent singular form, make first letter upper case:

```
5232       \glsabbrvfont{\Glsaccessshort{\glslabel}}\glsinsert
5233       }%
5234       {%
```

Subsequent singular form, all caps:

```
5235      \mfirstucMakeUppercase
5236      {\glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert}%
5237      }%
5238      }%
5239      }%
5240      {%
```

First use:

```
5241      \glsifplural
5242      {%
```

First use plural form:

```
5243      \glscapscase
5244      {%
```

First use plural form, don't adjust case:

```
5245      \glxtrfullplformat{\glslabel}{\glsinsert}%
5246      }%
5247      {%
```

First use plural form, make first letter upper case:

```
5248      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
5249      }%
5250      {%
```

First use plural form, all caps:

```
5251      \mfirstucMakeUppercase
5252      {\glxtrfullplformat{\glslabel}{\glsinsert}}%
5253      }%
5254      }%
5255      {%
```

First use singular form

```
5256      \glscapscase
5257      {%
```

First use singular form, don't adjust case:

```
5258      \glxtrfullformat{\glslabel}{\glsinsert}%
5259      }%
5260      {%
```

First use singular form, make first letter upper case:

```
5261      \Glsxtrfullformat{\glslabel}{\glsinsert}%
5262      }%
5263      {%
```

First use singular form, all caps:

```
5264      \mfirstucMakeUppercase
5265      {\glxtrfullformat{\glslabel}{\glsinsert}}%
5266      }%
5267      }%
5268      }%
```

```

5269 }%
5270 {%
    User supplied text.
5271 \glscustomtext
5272 }%
5273 }

```

1.6.1 Abbreviation Styles Setup

abbreviationstyle

```

5274 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
5275   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
5276   {%
5277     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
5278   }%
5279   {%

```

Have abbreviations already been defined for this category?

```

5280   \ifcsstring{@glsabbrv@current@#1}{#2}%
5281   {%

```

Style already set.

```

5282   }%
5283   {%
5284     \def\@glsxtr@dostylewarn{%
5285       \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
5286       {%
5287         \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
5288           style has been switched \MessageBreak
5289           for category ‘#1’, \MessageBreak
5290           but there have already been entries \MessageBreak
5291           defined for this category. Unwanted \MessageBreak
5292           side-effects may result}}%
5293       \@endfortrue
5294     }%
5295     \@glsxtr@dostylewarn

```

Set up the style for the given category.

```

5296     \csdef{@glsabbrv@current@#1}{#2}%
5297     \glsxtr@applyabbrvstyle{#2}%
5298   }%
5299 }%
5300 }

```

applyabbrvstyle Apply the abbreviation style without existence check.

```

5301 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
5302   \csuse{@glsabbrv@dispstyle@setup@#1}%
5303   \csuse{@glsabbrv@dispstyle@fmts@#1}%
5304 }

```

r@applyabbrvfmt Only apply the style formats.

```
5305 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
5306   \csuse{@glsabbrv@dispstyle@fmts@#1}%
5307 }
```

breivationstyle This is different from \newacronymstyle. The first argument is the label, the second argument sets the information required when defining the new abbreviation and the third argument sets the commands used to display the full format.

```
5308 \newcommand*{\newabbreviationstyle}[3]{%
5309   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
5310     {%
5311       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
5312         defined}{}%
5313     }%
5314     {%
5315       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
5316       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
5317       #2}%
5318       \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
5319       \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
5320       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
5321       \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
5322       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
5323       #3}%
5324     }%
5325 }
```

breivationstyle

```
5326 \newcommand*{\renewabbreviationstyle}[3]{%
5327   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
5328     {%
5329       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
5330     }%
5331     {%
5332       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
5333       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
5334       #2}%
5335       \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
5336       \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
5337       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
5338       \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
5339       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

```

5340     #3}%
5341   }%
5342 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

5343 \newcommand*{\letabbreviationstyle}[2]{%
5344   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
5345   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
5346 }

```

deprecated@abbrstyle `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

5347 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
5348   \csdef{@glsabbrv@dispstyle@setup@#1}{%
5349     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
5350     \csuse{@glsabbrv@dispstyle@setup@#2}%
5351   }%
5352   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
5353 }

```

deprecatedAbbrStyle Generate warning for deprecated style use.

```

5354 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
5355   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
5356     use ‘#2’ instead}%
5357 }

```

useAbbrStyleSetup

```

5358 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
5359   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
5360   {%
5361     \PackageError{glossaries-extra}%
5362       {Unknown abbreviation style definitions ‘#1’}{}%
5363   }%
5364   {%
5365     \csname @glsabbrv@dispstyle@setup@#1\endcsname
5366   }%
5367 }

```

useAbbrStyleFmt

```

5368 \newcommand*{\GlsXtrUseAbbrStyleFmt}[1]{%
5369   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
5370   {%
5371     \PackageError{glossaries-extra}%
5372       {Unknown abbreviation style formats ‘#1’}{}%

```

```

5373 }%
5374 {%
5375   \csname @glsabbrv@dispstyle@fmts@#1\endcsname
5376 }%
5377 }

```

1.6.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn't set to "true". If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glstrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

`xtrinsertinside` Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```

5378 \newif\ifglstrinsertinside
5379 \glstrinsertinsidefalse

```

`long-short`

```

5380 \newabbreviationstyle{long-short}%
5381 {%
5382   \renewcommand*{\CustomAbbreviationFields}{%
5383     name={\protect\glsabbrvfont{\the\glsshorttok}},
5384     sort={\the\glsshorttok},
5385     first={\protect\glsfirstlongfont{\the\glslongtok}}%
5386     \protect\glstrfullsep{\the\glslabeltok}}%
5387     (\protect\glsfirstabbrvfont{\the\glsshorttok}}),%
5388     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%
5389     \protect\glstrfullsep{\the\glslabeltok}}%
5390     (\protect\glsfirstabbrvfont{\the\glsshortpltok}}),%
5391     plural={\protect\glsabbvfont{\the\glsshortpltok}},%
5392     description={\the\glslongtok}}%

```

Unset the regular attribute if it has been set.

```

5393 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5394   \glshasattribute{\the\glslabeltok}{regular}%
5395   {%
5396     \glissetattribute{\the\glslabeltok}{regular}{false}%
5397   }%
5398   {}%
5399 }%
5400 }%
5401 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5402 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%

```

```

5403 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5404 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5405 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5406 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5407 \renewcommand*{\glsxtrfullformat}[2]{%
5408   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5409   \ifglsxtrininsertinside\else##2\fi
5410   \glsxtrfullsep{##1}%
5411   (\glsfirstabbrvfont{\glsaccessshort{##1}})%
5412 }%
5413 \renewcommand*{\glsxtrfullplformat}[2]{%
5414   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
5415   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5416   (\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
5417 }%
5418 \renewcommand*{\Glsxtrfullformat}[2]{%
5419   \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5420   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5421   (\glsfirstabbrvfont{\glsaccessshort{##1}})%
5422 }%
5423 \renewcommand*{\Glsxtrfullplformat}[2]{%
5424   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
5425   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5426   (\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
5427 }%
5428 }

```

Set this as the default style for general abbreviations:

```

5429 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

5430 \newcommand*{\glsxtrlongshortdescsort}{\the\glslongtok\space(\the\glsshorttok)}

```

long-short-desc

User supplies description. The long form is included in the name.

```

5431 \newabbreviationstyle{long-short-desc}%
5432 {%
5433   \renewcommand*{\CustomAbbreviationFields}{%
5434     name={\protect\glsxtrfullformat{\the\glslabeltok}{}},
5435     sort={\glsxtrlongshortdescsort},%
5436     first={\protect\glsfirstlongfont{\the\glslongtok}%
5437       \protect\glsxtrfullsep{\the\glslabeltok}%
5438       (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
5439     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
5440       \protect\glsxtrfullsep{\the\glslabeltok}%
5441       (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%

```

The text key should only have the short form.

```

5442   text={\protect\glsabbrvfont{\the\glsshorttok}},%

```

```

5443 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
5444 }%

Unset the regular attribute if it has been set.
5445 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5446   \glshasattribute{\the\glslabeltok}{regular}%
5447   {%
5448     \glissetattribute{\the\glslabeltok}{regular}{false}%
5449   }%
5450   {}%
5451 }%
5452 }%
5453 {%
5454   \GlsXtrUseAbbrStyleFmts{long-short}%
5455 }

```

short-long Short form followed by long form in parenthesis on first use.

```

5456 \newabbreviationstyle{short-long}%
5457 {%
5458   \renewcommand*{\CustomAbbreviationFields}{%
5459     name={\protect\glsabbrvfont{\the\glsshorttok}},
5460     sort={\the\glsshorttok},
5461     description={\the\glslongtok},%
5462     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
5463     \protect\glxtrfullsep{\the\glslabeltok}%
5464     (\protect\glsfirstlongfont{\the\glslongtok})),%
5465     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
5466     \protect\glxtrfullsep{\the\glslabeltok}%
5467     (\protect\glsfirstlongfont{\the\glslongpltok})),%
5468     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

Unset the regular attribute if it has been set.
5469 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5470   \glshasattribute{\the\glslabeltok}{regular}%
5471   {%
5472     \glissetattribute{\the\glslabeltok}{regular}{false}%
5473   }%
5474   {}%
5475 }%
5476 }%
5477 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5478 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
5479 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5480 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5481 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5482 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5483 \renewcommand*{\glxtrfullformat}[2]{%

```



```

5484 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5485 \ifglxtrinsertinside\else##2\fi
5486 \glxtrfullsep{##1}%
5487 (\glsfirstlongfont{\glsaccesslong{##1}})%
5488 }%
5489 \renewcommand*{\glxtrfullplformat}[2]{%
5490 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5491 \ifglxtrinsertinside\else##2\fi
5492 \glxtrfullsep{##1}%
5493 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
5494 }%
5495 \renewcommand*{\Glsxtrfullformat}[2]{%
5496 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5497 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5498 (\glsfirstlongfont{\glsaccesslong{##1}})%
5499 }%
5500 \renewcommand*{\Glsxtrfullplformat}[2]{%
5501 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5502 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5503 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
5504 }%
5505 }

```

short-long-desc User supplies description. The long form is included in the name.

```

5506 \newabbreviationstyle{short-long-desc}%
5507 {%
5508 \renewcommand*{\CustomAbbreviationFields}{%
5509 name={\protect\glxtrfullformat{\the\glslabeltok}{}},
5510 sort={\the\glsshorttok},%
5511 first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
5512 \protect\glxtrfullsep{\the\glslabeltok}%
5513 (\protect\glsfirstlongfont{\the\glslongtok})},%
5514 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
5515 \protect\glxtrfullsep{\the\glslabeltok}%
5516 (\protect\glsfirstlongfont{\the\glslongpltok})},%
5517 text={\protect\glsabbrvfont{\the\glsshorttok}},%
5518 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
5519 }%

```

Unset the regular attribute if it has been set.

```

5520 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5521 \glshasattribute{\the\glslabeltok}{regular}%
5522 {%
5523 \glssetattribute{\the\glslabeltok}{regular}{false}%
5524 }%
5525 {}%
5526 }%
5527 }%

```

```

5528 {%
5529   \GlsXtrUseAbbrStyleFmts{short-long}%
5530 }

```

`\longfootnotefont` Only used by the “footnote” styles.

```

5531 \newcommand*{\glslongfootnotefont}[1]{\glslongfootnotefont{#1}}%

```

`\longfootnotefont` Only used by the “footnote” styles.

```

5532 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%

```

`\glslxtrabbrvfootnote` `\glslxtrabbrvfootnote{<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`).

```

5533 \newcommand*{\glslxtrabbrvfootnote}[2]{\footnote{#2}}

```

`\footnote` Short form followed by long form in footnote on first use.

```

5534 \newabbreviationstyle{footnote}%
5535 {%
5536   \renewcommand*{\CustomAbbreviationFields}{%
5537     name={\protect\glsabbrvfont{\the\glsshorttok}},
5538     sort={\the\glsshorttok},
5539     description={\the\glslongtok},%

5540     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
5541     \protect\glslxtrabbrvfootnote{\the\glslabeltok}%
5542     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
5543     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
5544     \protect\glslxtrabbrvfootnote{\the\glslabeltok}%
5545     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
5546     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.

```

5547   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5548     \glsssetAttribute{\the\glslabeltok}{nohyperfirst}{true}%
5549     \glshasattribute{\the\glslabeltok}{regular}%
5550     {%
5551       \glsssetAttribute{\the\glslabeltok}{regular}{false}%
5552     }%
5553   }%
5554 }%
5555 }%
5556 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5557 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
5558 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5559 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5560 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
5561 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

5562 \renewcommand*{\glxtrfullformat}[2]{%
5563   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
5564   \ifglxtrininsertinside\else##2\fi
5565   \protect\glxtrabbrvfootnote{##1}%
5566   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
5567 }%
5568 \renewcommand*{\glxtrfullplformat}[2]{%
5569   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
5570   \ifglxtrininsertinside\else##2\fi
5571   \protect\glxtrabbrvfootnote{##1}%
5572   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
5573 }%
5574 \renewcommand*{\Glsxtrfullformat}[2]{%
5575   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
5576   \ifglxtrininsertinside\else##2\fi
5577   \protect\glxtrabbrvfootnote{##1}%
5578   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
5579 }%
5580 \renewcommand*{\Glsxtrfullplformat}[2]{%
5581   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
5582   \ifglxtrininsertinside\else##2\fi
5583   \protect\glxtrabbrvfootnote{##1}%
5584   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
5585 }%

```

The first use full form and the inline full form use the short (long) style.

```

5586 \renewcommand*{\glxtrininlinefullformat}[2]{%
5587   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
5588   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
5589   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
5590 }%
5591 \renewcommand*{\glxtrininlinefullplformat}[2]{%
5592   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
5593   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
5594   (\glsfirstlongfootnotefont{\glsaccesslongpl{##1}})%
5595 }%
5596 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
5597   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
5598   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
5599   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
5600 }%
5601 \renewcommand*{\Glsxtrininlinefullplformat}[2]{%

```

```

5602 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5603 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5604 (\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}})%
5605 }%
5606 }

```

short-footnote

```

5607 \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.

```

5608 \newabbreviationstyle{postfootnote}%
5609 {%
5610 \renewcommand*{\CustomAbbreviationFields}{%
5611 name={\protect\glsabbrvfont{\the\glsshorttok}},
5612 sort={\the\glsshorttok},
5613 description={\the\glslongtok},%
5614 first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
5615 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
5616 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.

```

5617 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5618 \csdef{glxtrpostlink\glscategorylabel}{%
5619 \glxtrifwasfirstuse
5620 }%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

5621 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
5622 {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
5623 }%
5624 {}%
5625 }%
5626 \glshasattribute{\the\glslabeltok}{regular}%
5627 {%
5628 \glissetattribute{\the\glslabeltok}{regular}{false}%
5629 }%
5630 {}%
5631 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

5632 \renewcommand*{\glxtrsetupfulldefs}{%
5633 \let\glxtrifwasfirstuse\@secondoftwo
5634 }%
5635 }%
5636 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5637 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
5638 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvdefaultfont{##1}}%
5639 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvdefaultfont{##1}}%
5640 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongfootnotefont{##1}}%
5641 \renewcommand*{\glsslongfont}[1]{\glsslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

5642 \renewcommand*{\glxtrfullformat}[2]{%
5643   \glssfirstabbrvfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
5644   \ifglxtrininsertinside\else##2\fi
5645 }%
5646 \renewcommand*{\glxtrfullplformat}[2]{%
5647   \glssfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
5648   \ifglxtrininsertinside\else##2\fi
5649 }%
5650 \renewcommand*{\Glsxtrfullformat}[2]{%
5651   \glssfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
5652   \ifglxtrininsertinside\else##2\fi
5653 }%
5654 \renewcommand*{\Glsxtrfullplformat}[2]{%
5655   \glssfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
5656   \ifglxtrininsertinside\else##2\fi
5657 }%

```

The first use full form and the inline full form use the short (long) style.

```

5658 \renewcommand*{\glxtrininlinefullformat}[2]{%
5659   \glssfirstabbrvfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
5660   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
5661   (\glssfirstlongfootnotefont{\glssaccesslong{##1}})%
5662 }%
5663 \renewcommand*{\glxtrininlinefullplformat}[2]{%
5664   \glssfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
5665   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
5666   (\glssfirstlongfootnotefont{\glssaccesslongpl{##1}})%
5667 }%
5668 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
5669   \glssfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
5670   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
5671   (\glssfirstlongfootnotefont{\Glsaccesslong{##1}})%
5672 }%
5673 \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
5674   \glssfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
5675   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
5676   (\glssfirstlongfootnotefont{\Glsaccesslongpl{##1}})%
5677 }%
5678 }

```

rt-postfootnote

```

5679 \letabbreviationstyle{short-postfootnote}{postfootnote}

```

short Provide a style that only displays the short form on first use, but the short and long form can be displayed with the “full” commands that use the inline format. If the user supplies a description, the long form won’t be displayed in the predefined glossary styles, but the post description hook can be employed to automatically insert it.

```

5680 \newabbreviationstyle{short}%
5681 {%
5682   \renewcommand*{\CustomAbbreviationFields}{%
5683     name={\protect\glsabbrvfont{\the\glsshorttok}},
5684     sort={\the\glsshorttok},
5685     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
5686     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
5687     text={\protect\glsabbrvfont{\the\glsshorttok}},
5688     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
5689     description={\the\glslongtok}}%
5690   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5691     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5692 }%
5693 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5694 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
5695 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5696 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5697 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5698 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

5699 \renewcommand*{\glsxtrinlinefullformat}[2]{%
5700   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
5701   \ifglsxtrininsertinside##2\fi}%
5702   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5703   (\glsfirstlongfont{\glsaccesslong{##1}})%
5704 }%
5705 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
5706   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
5707   \ifglsxtrininsertinside##2\fi}%
5708   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5709   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
5710 }%
5711 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
5712   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
5713   \ifglsxtrininsertinside##2\fi}%
5714   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5715   (\glsfirstlongfont{\Glsaccesslong{##1}})%
5716 }%
5717 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
5718   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
5719   \ifglsxtrininsertinside##2\fi}%
5720   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5721   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%

```

5722 }%

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.

```
5723 \renewcommand*{\glxtrfullformat}[2]{%
5724   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5725   \ifglxtrinsertinside\else##2\fi
5726 }%
5727 \renewcommand*{\glxtrfullplformat}[2]{%
5728   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5729   \ifglxtrinsertinside\else##2\fi
5730 }%
5731 \renewcommand*{\Glsxtrfullformat}[2]{%
5732   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5733   \ifglxtrinsertinside\else##2\fi
5734 }%
5735 \renewcommand*{\Glsxtrfullplformat}[2]{%
5736   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5737   \ifglxtrinsertinside\else##2\fi
5738 }%
5739 }
```

Set this as the default style for acronyms:

```
5740 \setabbreviationstyle[acronym]{short}
```

short-nolong

```
5741 \letabbreviationstyle{short-nolong}{short}
```

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.

```
5742 \newabbreviationstyle{short-desc}%
5743 {%
5744   \renewcommand*{\CustomAbbreviationFields}{%
5745     name={\protect\glxtrinlinefullformat{\the\glslabeltok}{}},
5746     sort={\the\glsshorttok},
5747     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
5748     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
5749     text={\protect\glsabbrvfont{\the\glsshorttok}},
5750     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
5751     description={\the\glslongtok}}%
5752   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5753     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5754 }%
5755 }
```

In case the user wants to mix and match font styles, these are redefined here.

```
5756 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
5757 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5758 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
```

```

5759 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5760 \renewcommand*\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

5761 \renewcommand*\glsxtrinlinefullformat}[2]{%
5762   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
5763   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5764   (\glsfirstlongfont{\glsaccesslong{##1}})%
5765 }%
5766 \renewcommand*\glsxtrinlinefullplformat}[2]{%
5767   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
5768   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5769   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
5770 }%
5771 \renewcommand*\Glsxtrinlinefullformat}[2]{%
5772   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
5773   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5774   (\glsfirstlongfont{\Glsaccesslong{##1}})%
5775 }%
5776 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
5777   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
5778   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5779   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
5780 }%

```

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.

```

5781 \renewcommand*\glsxtrfullformat}[2]{%
5782   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
5783   \ifglsxtrinsertinside\else##2\fi
5784 }%
5785 \renewcommand*\glsxtrfullplformat}[2]{%
5786   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
5787   \ifglsxtrinsertinside\else##2\fi
5788 }%
5789 \renewcommand*\Glsxtrfullformat}[2]{%
5790   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
5791   \ifglsxtrinsertinside\else##2\fi
5792 }%
5793 \renewcommand*\Glsxtrfullplformat}[2]{%
5794   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
5795   \ifglsxtrinsertinside\else##2\fi
5796 }%
5797 }

```

ort-nolong-desc

```

5798 \letabbreviationstyle{short-nolong-desc}{short-desc}

```

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.

```

5799 \newabbreviationstyle{long-desc}%
5800 {%
5801   \renewcommand*{\CustomAbbreviationFields}{%
5802     name={\protect\protect\glsfirstlongfont{\the\glslongtok}},
5803     sort={\the\glslongtok},
5804     first={\protect\glsfirstlongfont{\the\glslongtok}},
5805     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5806     text={\the\glslongtok},
5807     plural={\the\glslongpltok}%
5808   }%
5809   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5810     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5811 }%
5812 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5813 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
5814 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5815 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5816 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5817 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the long format followed by the short form in parentheses.

```

5818 \renewcommand*{\glsxtrinlinefullformat}[2]{%
5819   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5820   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5821   (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
5822 }%
5823 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
5824   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
5825   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5826   (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
5827 }%
5828 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
5829   \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5830   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5831   (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
5832 }%
5833 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
5834   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
5835   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5836   (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
5837 }%

```

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.

```

5838 \renewcommand*{\glsxtrfullformat}[2]{%
5839   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5840   \ifglsxtrininsertinside\else##2\fi

```

```

5841 }%
5842 \renewcommand*{\glxtrfullplformat}[2]{%
5843   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5844   \ifglxtrinsertinside\else##2\fi
5845 }%
5846 \renewcommand*{\Glsxtrfullformat}[2]{%
5847   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
5848   \ifglxtrinsertinside\else##2\fi
5849 }%
5850 \renewcommand*{\Glsxtrfullplformat}[2]{%
5851   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5852   \ifglxtrinsertinside\else##2\fi
5853 }%
5854 }

```

ng-noshort-desc Provide a synonym that matches similar styles.

```
5855 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

long It doesn't really make a great deal of sense to have a long-only style that doesn't have a description, but the best course of action here is to use the short form as the name and the long form as the description.

```

5856 \newabbreviationstyle{long}%
5857 {%
5858   \renewcommand*{\CustomAbbreviationFields}{%
5859     name={\protect\glsabbrvfont{\the\glsshorttok}},
5860     sort={\the\glsshorttok},
5861     first={\protect\glsfirstlongfont{\the\glslongtok}},
5862     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5863     text={\the\glslongtok},
5864     plural={\the\glslongpltok},%
5865     description={\the\glslongtok}%
5866   }%
5867   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5868     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5869 }%
5870 {%
5871   \GlsXtrUseAbbrStyleFmts{long-desc}%
5872 }

```

long-noshort Provide a synonym that matches similar styles.

```
5873 \letabbreviationstyle{long-noshort}{long}
```

1.6.3 Predefined Styles (Small Capitals)

These styles use:

\glxtrscfont

```
5874 \newcommand*{\glxtrscfont}[1]{\textsc{#1}}
```

sxtrfirstscfont

```
5875 \newcommand*{\glxtrfirstscfont}[1]{\glxtrscfont{#1}}
```

and for the default short form suffix:

\glxtrscsuffix

```
5876 \newcommand*{\glxtrscsuffix}{\glstextup{\glxtrabbrvpluralsuffix}}
```

long-short-sc

```
5877 \newabbreviationstyle{long-short-sc}%  
5878 {%  
5879   \GlsXtrUseAbbrStyleSetup{long-short}%  
5880 }%  
5881 {%
```

Mostly as long-short style:

```
5882   \GlsXtrUseAbbrStyleFmts{long-short}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5883   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5884   \renewcommand*{\glssabbrvfont}[1]{\glxtrscfont{##1}}%  
5885   \renewcommand*{\glssfirstabbrvfont}[1]{\glxtrfirstscfont{##1}}%  
5886 }
```

g-short-sc-desc

```
5887 \newabbreviationstyle{long-short-sc-desc}%  
5888 {%  
5889   \GlsXtrUseAbbrStyleSetup{long-short-desc}%  
5890 }%  
5891 {%
```

Mostly as long-short-desc style:

```
5892   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5893   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5894   \renewcommand*{\glssabbrvfont}[1]{\glxtrscfont{##1}}%  
5895   \renewcommand*{\glssfirstabbrvfont}[1]{\glxtrfirstscfont{##1}}%  
5896 }
```

Now the short (long) version

```
5897 \newabbreviationstyle{short-sc-long}%  
5898 {%  
5899   \GlsXtrUseAbbrStyleSetup{short-long}%  
5900 }%  
5901 {%
```

Mostly as short-long style:

```
5902   \GlsXtrUseAbbrStyleFmts{short-long}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5903 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5904 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5905 \renewcommand*{\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%
5906 }
```

As before but user provides description

```
5907 \newabbreviationstyle{short-sc-long-desc}%
5908 {%
5909 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5910 }%
5911 {%
```

Mostly as short-long-desc style:

```
5912 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5913 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5914 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5915 \renewcommand*{\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%
5916 }
```

short-sc

```
5917 \newabbreviationstyle{short-sc}%
5918 {%
5919 \GlsXtrUseAbbrStyleSetup{short-nolong}%
5920 }%
5921 {%
```

Mostly as short style:

```
5922 \GlsXtrUseAbbrStyleFmts{short-nolong}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5923 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5924 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5925 \renewcommand*{\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%
5926 }
```

short-sc-nolong

```
5927 \letabbreviationstyle{short-sc-nolong}{short-sc}
```

short-sc-desc

```
5928 \newabbreviationstyle{short-sc-desc}%
5929 {%
5930 \GlsXtrUseAbbrStyleSetup{short-desc}%
5931 }%
5932 {%
```

Mostly as short style:

```
5933 \GlsXtrUseAbbrStyleFmts{short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5934 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5935 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5936 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
5937 }
```

-sc-nolong-desc

```
5938 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}
```

long-noshort-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like \glssshort.

```
5939 \newabbreviationstyle{long-noshort-sc}%
5940 {%
5941 \GlsXtrUseAbbrStyleSetup{long-noshort}%
5942 }%
5943 {%
```

Mostly as long style:

```
5944 \GlsXtrUseAbbrStyleFmts{long-noshort}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5945 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5946 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5947 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
5948 }
```

long-sc Backward compatibility:

```
5949 \@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 \glssshort.

```
5950 \newabbreviationstyle{long-noshort-sc-desc}%
5951 {%
5952 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5953 }%
5954 {%
```

Mostly as long style:

```
5955 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5956 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5957 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5958 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
5959 }
```

long-desc-sc Backward compatibility:

```
5960 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}
```

ort-sc-footnote

```
5961 \newabbreviationstyle{short-sc-footnote}%  
5962 {%  
5963   \GlsXtrUseAbbrStyleSetup{short-footnote}%  
5964 }%  
5965 {%
```

Mostly as long style:

```
5966   \GlsXtrUseAbbrStyleFmts{short-footnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5967   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5968   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%  
5969   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%  
5970 }
```

footnote-sc Backward compatibility:

```
5971 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}
```

sc-postfootnote

```
5972 \newabbreviationstyle{short-sc-postfootnote}%  
5973 {%  
5974   \GlsXtrUseAbbrStyleSetup{short-postfootnote}%  
5975 }%  
5976 {%
```

Mostly as long style:

```
5977   \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5978   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5979   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%  
5980   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%  
5981 }
```

postfootnote-sc Backward compatibility:

```
5982 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}
```

1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the relsize package, which must be loaded by the user. These styles all use:

\glxtrsmfont

```
5983 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}
```

sxtrfirstsmfont

```
5984 \newcommand*{\glxtrfirstsmfont}[1]{\glxtrsmfont{#1}}
```

and for the default short form suffix:

\glxtrmsuffix

```
5985 \newcommand*{\glxtrmsuffix}{\glxtrabbrvpluralsuffix}
```

long-short-sm

```
5986 \newabbreviationstyle{long-short-sm}%
5987 {%
5988   \GlsXtrUseAbbrStyleSetup{long-short}%
5989 }%
5990 {%
  Mostly as long-short style:
5991   \GlsXtrUseAbbrStyleFmts{long-short}%
5992   \renewcommand*{\glsabbrvfont[1]{\glxtrsmfont{##1}}}%
5993   \renewcommand*{\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}}%
5994   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrmsuffix}%
5995 }
```

g-short-sm-desc

```
5996 \newabbreviationstyle{long-short-sm-desc}%
5997 {%
5998   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5999 }%
6000 {%
  Mostly as long-short-desc style:
6001   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
6002   \renewcommand*{\glsabbrvfont[1]{\glxtrsmfont{##1}}}%
6003   \renewcommand*{\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}}%
6004   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrmsuffix}%
6005 }
```

short-sm-long Now the short (long) version

```
6006 \newabbreviationstyle{short-sm-long}%
6007 {%
6008   \GlsXtrUseAbbrStyleSetup{short-long}%
6009 }%
6010 {%
  Mostly as short-long style:
6011   \GlsXtrUseAbbrStyleFmts{short-long}%
6012   \renewcommand*{\glsabbrvfont[1]{\glxtrsmfont{##1}}}%
6013   \renewcommand*{\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}}%
6014   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrmsuffix}%
6015 }
```

rt-sm-long-desc As before but user provides description

```
6016 \newabbreviationstyle{short-sm-long-desc}%
6017 {%
6018   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
6019 }%
6020 {%
```

Mostly as short-long-desc style:

```
6021 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
6022 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
6023 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
6024 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6025 }
```

short-sm

```
6026 \newabbreviationstyle{short-sm}%
6027 {%
6028 \GlsXtrUseAbbrStyleSetup{short-nolong}%
6029 }%
6030 {%
```

Mostly as short style:

```
6031 \GlsXtrUseAbbrStyleFmts{short-nolong}%
6032 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
6033 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
6034 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6035 }
```

short-sm-nolong

```
6036 \letabbreviationstyle{short-sm-nolong}{short-sm}
```

short-sm-desc

```
6037 \newabbreviationstyle{short-sm-desc}%
6038 {%
6039 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
6040 }%
6041 {%
```

Mostly as short style:

```
6042 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
6043 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
6044 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
6045 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6046 }
```

-sm-nolong-desc

```
6047 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}
```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like \glsshort.

```
6048 \newabbreviationstyle{long-noshort-sm}%
6049 {%
6050 \GlsXtrUseAbbrStyleSetup{long-noshort}%
6051 }%
6052 {%
```


Mostly as long style:

```
6053 \GlsXtrUseAbbrStyleFmts{long-noshort}%
6054 \renewcommand*\glsabrvfont[1]{\glxtrsmfont{##1}}%
6055 \renewcommand*\glsfirstabrvfont[1]{\glxtrfirstsmfont{##1}}%
6056 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6057 }
```

long-sm Backward compatibility:

```
6058 \@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.

```
6059 \newabbreviationstyle{long-noshort-sm-desc}%
6060 {%
6061 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
6062 }%
6063 {%
```

Mostly as long style:

```
6064 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
6065 \renewcommand*\glsabrvfont[1]{\glxtrsmfont{##1}}%
6066 \renewcommand*\glsfirstabrvfont[1]{\glxtrfirstsmfont{##1}}%
6067 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6068 }
```

long-desc-sm Backward compatibility:

```
6069 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}
```

short-sm-footnote

```
6070 \newabbreviationstyle{short-sm-footnote}%
6071 {%
6072 \GlsXtrUseAbbrStyleSetup{short-footnote}%
6073 }%
6074 {%
```

Mostly as long style:

```
6075 \GlsXtrUseAbbrStyleFmts{short-footnote}%
6076 \renewcommand*\glsabrvfont[1]{\glxtrsmfont{##1}}%
6077 \renewcommand*\glsfirstabrvfont[1]{\glxtrfirstsmfont{##1}}%
6078 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6079 }
```

footnote-sm Backward compatibility:

```
6080 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}
```

sm-postfootnote

```
6081 \newabbreviationstyle{short-sm-postfootnote}%
6082 {%
6083 \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
```

6084 }%

6085 {%

Mostly as long style:

6086 \GlsXtrUseAbbrStyleFmts{short-postfootnote}%

6087 \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%

6088 \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}%

6089 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%

6090 }

postfootnote-sm Backward compatibility:

6091 \@glstr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

1.6.5 Predefined Styles (Emphasized)

These styles use \emph for the short form.

\glsabbrvemfont

6092 \newcommand*\glsabbrvemfont[1]{\emph{#1}}%

irstabbrvemfont

6093 \newcommand*\glsfirstabbrvemfont[1]{\glsabbrvemfont{#1}}%

firstlongemfont Only used by the “long-em” styles.

6094 \newcommand*\glsfirstlongemfont[1]{\glslongemfont{#1}}%

\glslongemfont Only used by the “long-em” styles.

6095 \newcommand*\glslongemfont[1]{\emph{#1}}%

long-short-em

6096 \newabbreviationstyle{long-short-em}%

6097 {%

6098 \GlsXtrUseAbbrStyleSetup{long-short}%

6099 }%

6100 {%

Mostly as long-short style:

6101 \GlsXtrUseAbbrStyleFmts{long-short}%

6102 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%

6103 }

g-short-em-desc

6104 \newabbreviationstyle{long-short-em-desc}%

6105 {%

6106 \GlsXtrUseAbbrStyleSetup{long-short-desc}%

6107 }%

6108 {%

Mostly as long-short-desc style:

```
6109 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
6110 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6111 }
```

ong-em-short-em

```
6112 \newabbreviationstyle{long-em-short-em}%
6113 {%
```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```
6114 \renewcommand*{\CustomAbbreviationFields}{%
6115   name={\protect\glsabbrvfont{\the\glsshorttok}},
6116   sort={\the\glsshorttok},
6117   first={\protect\glsfirstlongfont{\the\glslongtok}%
6118     \protect\glsxtrfullsep{\the\glslabeltok}%
6119     (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
6120   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6121     \protect\glsxtrfullsep{\the\glslabeltok}%
6122     (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%
6123   plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6124   description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```
6125 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6126   \glsattribute{\the\glslabeltok}{regular}%
6127   {%
6128     \glssetattribute{\the\glslabeltok}{regular}{false}%
6129   }%
6130   {}%
6131 }%
6132 }%
6133 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6134 \GlsXtrUseAbbrStyleFmts{long-short}%
6135 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6136 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
6137 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
6138 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
6139 }
```

m-short-em-desc

```
6140 \newabbreviationstyle{long-em-short-em-desc}%
6141 {%
6142   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
6143 }%
6144 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6145 \GlsXtrUseAbbrStyleFmts{long-short-desc}%

```

```

6146 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6147 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
6148 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
6149 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
6150 }

```

short-em-long Now the short (long) version

```

6151 \newabbreviationstyle{short-em-long}%
6152 {%
6153   \GlsXtrUseAbbrStyleSetup{short-long}%
6154 }%
6155 {%

```

Mostly as short-long style:

```

6156 \GlsXtrUseAbbrStyleFmts{short-long}%
6157 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6158 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
6159 }

```

short-em-long-desc As before but user provides description

```

6160 \newabbreviationstyle{short-em-long-desc}%
6161 {%
6162   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
6163 }%
6164 {%

```

Mostly as short-long-desc style:

```

6165 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
6166 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6167 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
6168 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
6169 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
6170 }

```

short-em-long-em

```

6171 \newabbreviationstyle{short-em-long-em}%
6172 {%

```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```

6173 \renewcommand*{\CustomAbbreviationFields}{%
6174   name={\protect\glsabbrvfont{\the\glsshorttok}},
6175   sort={\the\glsshorttok},
6176   description={\protect\glslongemfont{\the\glslongtok}},%
6177   first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6178     {\protect\glsxtrfullsep{\the\glslabeltok}}%
6179     ({\protect\glsfirstlongfont{\the\glslongtok}}),%
6180   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6181     {\protect\glsxtrfullsep{\the\glslabeltok}}%
6182     ({\protect\glsfirstlongfont{\the\glslongpltok}}),%
6183   plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```
6184 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6185   \glshasattribute{\the\glslabeltok}{regular}%
6186   {%
6187     \glsetattribute{\the\glslabeltok}{regular}{false}%
6188   }%
6189   {}%
6190 }%
6191 }%
6192 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6193 \GlsXtrUseAbbrStyleFmts{short-long}%
6194 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
6195 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
6196 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
6197 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
6198 }
```

em-long-em-desc

```
6199 \newabbreviationstyle{short-em-long-em-desc}%
6200 {%
6201   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
6202 }%
6203 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6204 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
6205 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
6206 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
6207 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
6208 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
6209 }
```

short-em

```
6210 \newabbreviationstyle{short-em}%
6211 {%
6212   \GlsXtrUseAbbrStyleSetup{short-nolong}%
6213 }%
6214 {%
```

Mostly as short style:

```
6215 \GlsXtrUseAbbrStyleFmts{short-nolong}%
6216 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
6217 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
6218 }
```

short-em-nolong

```
6219 \letabbreviationstyle{short-em-nolong}{short-em}
```

```

short-em-desc
6220 \newabbreviationstyle{short-em-desc}%
6221 {%
6222   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
6223 }%
6224 {%

    Mostly as short style:
6225   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
6226   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6227   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6228 }

-em-nolong-desc
6229 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

long-noshort-em  The short form is explicitly invoked through commands like \glsshort.
6230 \newabbreviationstyle{long-noshort-em}%
6231 {%
6232   \GlsXtrUseAbbrStyleSetup{long-noshort}%
6233 }%
6234 {%

    Mostly as long-noshort style:
6235   \GlsXtrUseAbbrStyleFmts{long-noshort}%
6236   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6237   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6238 }

long-em  Backward compatibility:
6239 \@glsxtr@deprecated@abbrstyle{long-em}{long-noshort-em}

g-em-noshort-em  The short form is explicitly invoked through commands like \glsshort.
6240 \newabbreviationstyle{long-em-noshort-em}%
6241 {%
6242   \renewcommand*\CustomAbbreviationFields{%
6243     name={\protect\glsabbrvfont{\the\glsshorttok}},
6244     sort={\the\glsshorttok},
6245     first={\protect\glsfirstlongfont{\the\glslongtok}},
6246     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
6247     text={\the\glslongtok},
6248     plural={\the\glslongpltok},%
6249     description={\protect\glslongemfont{\the\glslongtok}}%
6250   }%
6251   \renewcommand*\GlsXtrPostNewAbbreviation{%
6252     \glssetattribute{\the\glslabeltok}{regular}{true}}%
6253 }%
6254 {%

```

Mostly as long-noshort style:

```
6255 \GlsXtrUseAbbrStyleFmts{long-noshort}%
6256 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6257 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6258 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
6259 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
6260 }
```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like \glsshort.

```
6261 \newabbreviationstyle{long-noshort-em-desc}%
6262 {%
6263 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
6264 }%
6265 {%
```

Mostly as long style:

```
6266 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
6267 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6268 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6269 }
```

long-desc-em Backward compatibility:

```
6270 \@glsxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}
```

noshort-em-desc The short form is explicitly invoked through commands like \glsshort. The long form is emphasized.

```
6271 \newabbreviationstyle{long-em-noshort-em-desc}%
6272 {%
6273 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
6274 }%
6275 {%
```

Mostly as long style:

```
6276 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
6277 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6278 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6279 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
6280 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
6281 }
```

ort-em-footnote

```
6282 \newabbreviationstyle{short-em-footnote}%
6283 {%
6284 \GlsXtrUseAbbrStyleSetup{short-footnote}%
6285 }%
6286 {%
```

Mostly as long style:

```
6287 \GlsXtrUseAbbrStyleFmts{short-footnote}%
6288 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6289 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6290 }
```

footnote-em Backward compatibility:

```
6291 \@glsxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}
```

em-postfootnote

```
6292 \newabbreviationstyle{short-em-postfootnote}%
6293 {%
6294 \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
6295 }%
6296 {%
```

Mostly as long style:

```
6297 \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
6298 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6299 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6300 }
```

postfootnote-em Backward compatibility:

```
6301 \@glsxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}
```

1.6.6 Predefined Styles (User Parentheses Hook)

These styles allow the user to adjust the parenthetical forms. These styles all test for the existence of the field given by:

glsxtruserfield Default is the useri field.

```
6302 \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.

```
6303 \ifdef\glscurrentfieldvalue
6304 {
6305 \newcommand*\glsxtruserparen[2]{%
6306 \glsxtrfullsep{#2}%
6307 (#1\ifglshasfield{\glsxtruserfield}{#2}{, \glscurrentfieldvalue}{})%
6308 }
6309 }
6310 {
6311 \newcommand*\glsxtruserparen[2]{%
6312 \glsxtrfullsep{#2}%
6313 (#1\ifglshasfield{\glsxtruserfield}{#2}{, \@glo@thisvalue}{})%
6314 }
6315 }
```


Font used for short form:

lsabbrvuserfont

```
6316 \newcommand*{\glsabbrvuserfont}[1]{#1}
```

Font used for short form on first use:

stabbrvuserfont

```
6317 \newcommand*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}
```

Font used for long form:

glslonguserfont

```
6318 \newcommand*{\glslonguserfont}[1]{#1}
```

Font used for long form on first use:

rstlonguserfont

```
6319 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
6320 \newcommand*{\glsxtrusersuffix}{\glsxtrabbrvpluralsuffix}
```

long-short-user

```
6321 \newabbreviationstyle{long-short-user}%
6322 {%
```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```
6323   \renewcommand*{\CustomAbbreviationFields}{%
6324     name={\protect\glsabbrvfont{\the\glsshorttok}},
6325     sort={\the\glsshorttok},
6326     first={\protect\glsfirstlongfont{\the\glslongtok}}%
6327     \protect\glsxtruserparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}{\the\glslabeltok}},
6328     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%
6329     \protect\glsxtruserparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}{\the\glslabeltok}},
6330     plural={\protect\glsabbbfont{\the\glsshortpltok}},%
6331     description={\protect\glslonguserfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
6332   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6333     \glshasattribute{\the\glslabeltok}{regular}%
6334     {%
6335       \glssetattribute{\the\glslabeltok}{regular}{false}%
6336     }%
6337   }%
6338 }%
6339 }%
6340 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6341 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
6342 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
6343 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
6344 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
6345 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
6346 \renewcommand*{\glxtrfullformat}[2]{%
6347   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6348   \ifglxtrininsertinside\else##2\fi
6349   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
6350 }%
6351 \renewcommand*{\glxtrfullplformat}[2]{%
6352   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6353   \ifglxtrininsertinside\else##2\fi
6354   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
6355 }%
6356 \renewcommand*{\Glsxtrfullformat}[2]{%
6357   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6358   \ifglxtrininsertinside\else##2\fi
6359   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
6360 }%
6361 \renewcommand*{\Glsxtrfullplformat}[2]{%
6362   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6363   \ifglxtrininsertinside\else##2\fi
6364   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
6365 }%
6366 }
```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```
6367 \newabbreviationstyle{long-postshort-user}%
6368 {%
6369   \renewcommand*{\CustomAbbreviationFields}{%
6370     name={\protect\glsabbrvfont{\the\glsshorttok}},
6371     sort={\the\glsshorttok},
6372     first={\protect\glsfirstlongfont{\the\glslongtok}},%
6373     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},%
6374     plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6375     description={\protect\glslonguserfont{\the\glslongtok}}}%
6376   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6377     \csdef{glxtrpostlink\glscategorylabel}{%
6378       \glxtrifwasfirstuse
6379       {%
6380         \glxtruserparen
6381         {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
6382         {\glslabel}%
6383       }%
6384     }%
```

```

6385 }%
6386 \glshasattribute{\the\glslabeltok}{regular}%
6387 {%
6388     \glissetattribute{\the\glslabeltok}{regular}{false}%
6389 }%
6390 {}%
6391 }%
6392 }%
6393 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6394 \renewcommand*\abbrvpluralsuffix{\glsxtrusersuffix}%
6395 \renewcommand*\glsabbrvfont[1]{\glsabbrvuserfont{##1}}%
6396 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvuserfont{##1}}%
6397 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonguserfont{##1}}%
6398 \renewcommand*\glslongfont[1]{\glslonguserfont{##1}}%

```

First use full form:

```

6399 \renewcommand*\glsxtrfullformat[2]{%
6400     \glsfirstlongfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6401     \ifglsxtrininsertinside\else##2\fi
6402 }%
6403 \renewcommand*\glsxtrfullplformat[2]{%
6404     \glsfirstlongfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6405     \ifglsxtrininsertinside\else##2\fi
6406 }%
6407 \renewcommand*\Glsxtrfullformat[2]{%
6408     \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6409     \ifglsxtrininsertinside\else##2\fi
6410 }%
6411 \renewcommand*\Glsxtrfullplformat[2]{%
6412     \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6413     \ifglsxtrininsertinside\else##2\fi
6414 }%

```

In-line format:

```

6415 \renewcommand*\glsxtrinlinefullformat[2]{%
6416     \glsfirstlongfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6417     \ifglsxtrininsertinside\else##2\fi
6418     \glsxtruserparen{\glsfirstabbrvfont{\glssaccessshort{##1}}}{##1}%
6419 }%
6420 \renewcommand*\glsxtrinlinefullplformat[2]{%
6421     \glsfirstlongfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6422     \ifglsxtrininsertinside\else##2\fi
6423     \glsxtruserparen{\glsfirstabbrvfont{\glssaccessshortpl{##1}}}{##1}%
6424 }%
6425 \renewcommand*\Glsxtrinlinefullformat[2]{%
6426     \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6427     \ifglsxtrininsertinside\else##2\fi
6428     \glsxtruserparen{\glsfirstabbrvfont{\glssaccessshort{##1}}}{##1}%
6429 }%

```

```

6430 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6431   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6432   \ifglxtrinsertinside\else##2\fi
6433   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
6434 }%
6435 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

6436 \newabbreviationstyle{long-postshort-user}%
6437 {%
6438   \renewcommand*{\CustomAbbreviationFields}{%
6439     name={\protect\glsfirstlongfont{\the\glslongtok}%
6440       \protect\glxtruserparen
6441         {\protect\glsabbrvfont{\the\glsshorttok}}{\the\glslabeltok}},
6442     sort={\the\glslongtok},
6443     first={\protect\glsfirstlongfont{\the\glslongtok}},%
6444     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},%
6445     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6446 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6447   \csdef{glxtrpostlink\glscategorylabel}{%
6448     \glxtrifwasfirstuse
6449     {%
6450       \glxtruserparen
6451         {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
6452         {\glslabel}%
6453     }%
6454   }%
6455 }%
6456 \glshasattribute{\the\glslabeltok}{regular}%
6457 {%
6458   \glsssetAttribute{\the\glslabeltok}{regular}{false}%
6459 }%
6460 {}%
6461 }%
6462 }%
6463 {%
6464   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
6465 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

6466 \newabbreviationstyle{short-postlong-user}%
6467 {%
6468   \renewcommand*{\CustomAbbreviationFields}{%
6469     name={\protect\glsabbrvfont{\the\glsshorttok}},
6470     sort={\the\glsshorttok},
6471     first={\protect\glsfirstlongfont{\the\glslongtok}},%
6472     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},%
6473     plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6474     description={\protect\glslonguserfont{\the\glslongtok}}}%

```

```

6475 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6476   \csdef{glxstrpostlink\glscategorylabel}{%
6477     \glxstrifwasfirstuse
6478     {%
6479       \glxtruserparen
6480       {\glsfirstabbrvuserfont{\glsentrylong{\glslabel}}}%
6481       {\glslabel}%
6482     }%
6483   }%
6484 }%
6485 \glshasattribute{\the\glslabeltok}{regular}%
6486 {%
6487   \glissetattribute{\the\glslabeltok}{regular}{false}%
6488 }%
6489 {}%
6490 }%
6491 }%
6492 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6493 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
6494 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvuserfont{##1}}%
6495 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
6496 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
6497 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

6498 \renewcommand*{\glxtrfullformat}[2]{%
6499   \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6500   \ifglxtrininsertinside\else##2\fi
6501 }%
6502 \renewcommand*{\glxtrfullplformat}[2]{%
6503   \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6504   \ifglxtrininsertinside\else##2\fi
6505 }%
6506 \renewcommand*{\Glsxtrfullformat}[2]{%
6507   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6508   \ifglxtrininsertinside\else##2\fi
6509 }%
6510 \renewcommand*{\Glsxtrfullplformat}[2]{%
6511   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6512   \ifglxtrininsertinside\else##2\fi
6513 }%

```

In-line format:

```

6514 \renewcommand*{\glxtrinlinefullformat}[2]{%
6515   \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6516   \ifglxtrininsertinside\else##2\fi
6517   \glxtruserparen{\glsfirstlongfont{\glssaccesslong{##1}}}{##1}%
6518 }%
6519 \renewcommand*{\glxtrinlinefullplformat}[2]{%

```

```

6520 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6521 \ifglxtrinsertinside\else##2\fi
6522 \glxtruserparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}{##1}%
6523 }%
6524 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6525 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6526 \ifglxtrinsertinside\else##2\fi
6527 \glxtruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%
6528 }%
6529 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6530 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6531 \ifglxtrinsertinside\else##2\fi
6532 \glxtruserparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}{##1}%
6533 }%
6534 }

```

`long-user-desc` Like short-postlong-user but leaves the user to specify the description.

```

6535 \newabbreviationstyle{short-postlong-user-desc}%
6536 {%
6537 \renewcommand*{\CustomAbbreviationFields}{%
6538   name={\protect\glsabbrvfont{\the\glsshorttok}%
6539     \protect\glxtruserparen
6540       {\protect\glsfirstlongfont{\the\glslongpltok}}%
6541       {\the\glslabeltok}},
6542   sort={\the\glsshorttok},
6543   first={\protect\glsfirstlongfont{\the\glslongtok}},%
6544   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},%
6545   plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6546 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6547 \csdef{glxtrpostlink\glscategorylabel}{%
6548   \glxtrifwasfirstuse
6549   {%
6550     \glxtruserparen
6551     {\glsfirstabbrvuserfont{\glsentrylong{\glslabel}}}%
6552     {\glslabel}%
6553   }%
6554   }%
6555 }%
6556 \glshasattribute{\the\glslabeltok}{regular}%
6557 {%
6558   \glissetattribute{\the\glslabeltok}{regular}{false}%
6559 }%
6560 {}%
6561 }%
6562 }%
6563 {%
6564 \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
6565 }

```

short-user-desc

```
6566 \newabbreviationstyle{long-short-user-desc}%
6567 {%
6568   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
6569 }%
6570 {%
6571   \GlsXtrUseAbbrStyleFmts{long-short-user}%
6572 }
```

short-long-user

```
6573 \newabbreviationstyle{short-long-user}%
6574 {%
  \glslonguserfont is used in the description since \glsdesc doesn't set the style.
6575   \renewcommand*{\CustomAbbreviationFields}{%
6576     name={\protect\glsabbrvfont{\the\glsshorttok}},
6577     sort={\the\glsshorttok},
6578     description={\protect\glslonguserfont{\the\glslongtok}},%
6579     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
6580     \protect\glsextruserparen{\protect\glslonguserfont{\the\glslongtok}}{\the\glslabeltok}},%
6581     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
6582     \protect\glsextruserparen{\protect\glslonguserfont{\the\glslongpltok}}{\the\glslabeltok}},%
6583     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
  \Unset the regular attribute if it has been set.
6584   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6585     \glschasattribute{\the\glslabeltok}{regular}%
6586     {%
6587       \glssetattribute{\the\glslabeltok}{regular}{false}%
6588     }%
6589     {}%
6590   }%
6591 }%
6592 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6593 \renewcommand*{\abbrvpluralsuffix}{\glsextrusersuffix}%
6594 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
6595 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
6596 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
6597 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
6598 \renewcommand*{\glsextrfullformat}[2]{%
6599   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsextrinsertinside##2\fi}%
6600   \ifglsextrinsertinside\else##2\fi
6601   \glsextruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%
6602 }%
6603 \renewcommand*{\glsextrfullplformat}[2]{%
6604   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsextrinsertinside##2\fi}%
6605   \ifglsextrinsertinside\else##2\fi
```

```

6606 \glstruserparen{\glstrfirstlongfont{\glstraccesslongpl{##1}}}{##1}%
6607 }%
6608 \renewcommand*{\Glsxtrfullformat}[2]{%
6609 \glstrfirstabbrvfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
6610 \ifglstrinsertinside\else##2\fi
6611 \glstruserparen{\glstrfirstlongfont{\glstraccesslong{##1}}}{##1}%
6612 }%
6613 \renewcommand*{\Glsxtrfullplformat}[2]{%
6614 \glstrfirstabbrvfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
6615 \ifglstrinsertinside\else##2\fi
6616 \glstruserparen{\glstrfirstlongfont{\glstraccesslongpl{##1}}}{##1}%
6617 }%
6618 }

```

-long-user-desc

```

6619 \newabbreviationstyle{short-long-user-desc}%
6620 {%
6621 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
6622 }%
6623 {%
6624 \GlsXtrUseAbbrStyleFmts{short-long-user}%
6625 }

```

1.7 Using Entries in Headings

There are four main problems with using entries in sectioning commands: they can mess with the first use flag if they end up in the table of contents, they can add unwanted numbers to the entry's location list, the label is corrupted if used inside `\MakeUppercase` (which is used by the default headings style) and they need to be expandable for PDF bookmarks. The glossaries package therefore recommends the use of the expandable commands, such as `\glstryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glstrabbrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref`'s `\texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glstrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to false, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that glossaries automatically loads `textcase`, so the label can be protected from case change with `textcase`'s `\NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
6626 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
6627 \renewcommand*{\markright}[1]{%
```

```
6628   \glsxtrmarkhook
```

```
6629   \@glsxtr@org@markright{\@glsxtrinmark#1\@glsxtrnotinmark}%
```

```
6630   \glsxtrrestoremarkhook
```

```
6631 }
```

`\markboth` Save original definition:

```
6632 \let\@glsxtr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
6633 \renewcommand*{\markboth}[2]{%
```

```
6634   \glsxtrmarkhook
```

```
6635   \@glsxtr@org@markboth
```

```
6636     {\@glsxtrinmark#1\@glsxtrnotinmark}%
```

```
6637     {\@glsxtrinmark#2\@glsxtrnotinmark}%
```

```
6638   \glsxtrrestoremarkhook
```

```
6639 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

`sxtrRevertMarks`

```
6640 \newcommand*{\glsxtrRevertMarks}{%
```

```
6641   \let\markright\@glsxtr@org@markright
```

```
6642   \let\markboth\@glsxtr@org@markboth
```

```
6643 }
```

`\glsxtrifinmark`

```
6644 \newcommand*{\glsxtrifinmark}[2]{#2}
```

`\@glsxtrinmark`

```
6645 \newrobustcmd*{\@glsxtrinmark}{%
```

```
6646   \let\glsxtrifinmark\@firstoftwo
```

```
6647 }
```

`glsxtrnotinmark`

```
6648 \newrobustcmd*{\@glsxtrnotinmark}{%
```

```
6649   \let\glsxtrifinmark\@secondoftwo
```

```
6650 }
```

`\glsxtrmarkhook` Hook used in new definition of `\markboth` and `\markright` to make some changes to apply to the marks:

```
6651 \newcommand*{\glsxtrmarkhook}{%
```

Save current definitions:

```

6652 \let\@glsxtr@org@MakeUppercase\MakeUppercase
6653 \let\@glsxtr@org@glsxtrtitleshort\glsxtrtitleshort
6654 \let\@glsxtr@org@glsxtrtitleshortpl\glsxtrtitleshortpl
6655 \let\@glsxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
6656 \let\@glsxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
6657 \let\@glsxtr@org@glsxtrtitletext\glsxtrtitletext
6658 \let\@glsxtr@org@Glsxtrtitletext\Glsxtrtitletext
6659 \let\@glsxtr@org@glsxtrtitleplural\glsxtrtitleplural
6660 \let\@glsxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
6661 \let\@glsxtr@org@glsxtrtitlefirst\glsxtrtitlefirst
6662 \let\@glsxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
6663 \let\@glsxtr@org@glsxtrtitlefirstplural\glsxtrtitlefirstplural
6664 \let\@glsxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
6665 \let\@glsxtr@org@glsxtrtitlelong\glsxtrtitlelong
6666 \let\@glsxtr@org@glsxtrtitlelongpl\glsxtrtitlelongpl
6667 \let\@glsxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
6668 \let\@glsxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
6669 \let\@glsxtr@org@glsxtrtitlefull\glsxtrtitlefull
6670 \let\@glsxtr@org@glsxtrtitlefullpl\glsxtrtitlefullpl
6671 \let\@glsxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
6672 \let\@glsxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl

```

New definitions

```

6673 \let\glsxtrifinmark\@firstoftwo
6674 \let\MakeUppercase\MakeTextUppercase
6675 \let\glsxtrtitleshort\glsxtrheadshort
6676 \let\glsxtrtitleshortpl\glsxtrheadshortpl
6677 \let\Glsxtrtitleshort\Glsxtrheadshort
6678 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
6679 \let\glsxtrtitletext\glsxtrheadtext
6680 \let\Glsxtrtitletext\Glsxtrheadtext
6681 \let\glsxtrtitleplural\glsxtrheadplural
6682 \let\Glsxtrtitleplural\Glsxtrheadplural
6683 \let\glsxtrtitlefirst\glsxtrheadfirst
6684 \let\Glsxtrtitlefirst\Glsxtrheadfirst
6685 \let\glsxtrtitlefirstplural\glsxtrheadfirstplural
6686 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
6687 \let\glsxtrtitlelong\glsxtrheadlong
6688 \let\glsxtrtitlelongpl\glsxtrheadlongpl
6689 \let\Glsxtrtitlelong\Glsxtrheadlong
6690 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
6691 \let\glsxtrtitlefull\glsxtrheadfull
6692 \let\glsxtrtitlefullpl\glsxtrheadfullpl
6693 \let\Glsxtrtitlefull\Glsxtrheadfull
6694 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
6695 }

```

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.)

```

6696 \newcommand*{\glxtrrestoremarkhook}{%
6697   \let\glxtrifinmark\@secondoftwo
6698   \let\MakeUppercase\@glxtr@org@MakeUppercase
6699   \let\glxtrtitleshort\@glxtr@org@glxtrtitleshort
6700   \let\glxtrtitleshortpl\@glxtr@org@glxtrtitleshortpl
6701   \let\Glsxtrtitleshort\@glxtr@org@Glsxtrtitleshort
6702   \let\Glsxtrtitleshortpl\@glxtr@org@Glsxtrtitleshortpl
6703   \let\glxtrtitletext\@glxtr@org@glxtrtitletext
6704   \let\Glsxtrtitletext\@glxtr@org@Glsxtrtitletext
6705   \let\glxtrtitleplural\@glxtr@org@glxtrtitleplural
6706   \let\Glsxtrtitleplural\@glxtr@org@Glsxtrtitleplural
6707   \let\glxtrtitlefirst\@glxtr@org@glxtrtitlefirst
6708   \let\Glsxtrtitlefirst\@glxtr@org@Glsxtrtitlefirst
6709   \let\glxtrtitlefirstplural\@glxtr@org@glxtrtitlefirstplural
6710   \let\Glsxtrtitlefirstplural\@glxtr@org@Glsxtrtitlefirstplural
6711   \let\glxtrtitlelong\@glxtr@org@glxtrtitlelong
6712   \let\glxtrtitlelongpl\@glxtr@org@glxtrtitlelongpl
6713   \let\Glsxtrtitlelong\@glxtr@org@Glsxtrtitlelong
6714   \let\Glsxtrtitlelongpl\@glxtr@org@Glsxtrtitlelongpl
6715   \let\glxtrtitlefull\@glxtr@org@glxtrtitlefull
6716   \let\glxtrtitlefullpl\@glxtr@org@glxtrtitlefullpl
6717   \let\Glsxtrtitlefull\@glxtr@org@Glsxtrtitlefull
6718   \let\Glsxtrtitlefullpl\@glxtr@org@Glsxtrtitlefullpl
6719 }

```

Instead of using one document-wide conditional, use headuc attribute to determine whether or not to use the all upper case form.

glxtrheadshort Command used to display short form in the page header.

```

6720 \newcommand*{\glxtrheadshort}[1]{%
6721   \protect\NoCaseChange
6722   {%
6723     \gl@ifattribute{#1}{headuc}{true}%
6724     {%
6725       \GLSxtrshort[noindex,hyper=false]{#1}[]%
6726     }%
6727     {%
6728       \glxtrshort[noindex,hyper=false]{#1}[]%
6729     }%
6730   }%
6731 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

6732 \newrobustcmd*{\lsxtrtitleshort}[1]{%
6733   \glxtrshort[noindex,hyper=false]{#1}[]%
6734 }

```

`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.

```

6735 \newcommand*{\glsxtrheadshortpl}[1]{%
6736   \protect\NoCaseChange
6737   {%
6738     \glusifattribute{#1}{headuc}{true}%
6739     {%
6740       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
6741     }%
6742     {%
6743       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
6744     }%
6745   }%
6746 }

```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents.

```

6747 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
6748   \glsxtrshortpl[noindex,hyper=false]{#1}[]%
6749 }

```

`Glsxtrheadshort` Command used to display short form in the page header with the first letter converted to upper case.

```

6750 \newcommand*{\Glsxtrheadshort}[1]{%
6751   \protect\NoCaseChange
6752   {%
6753     \glusifattribute{#1}{headuc}{true}%
6754     {%
6755       \GLSxtrshort[noindex,hyper=false]{#1}[]%
6756     }%
6757     {%
6758       \Glsxtrshort[noindex,hyper=false]{#1}[]%
6759     }%
6760   }%
6761 }

```

`lxsxtrtitleshort` Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

6762 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
6763   \Glsxtrshort[noindex,hyper=false]{#1}[]%
6764 }

```

`sxtrheadshortpl` Command used to display plural short form in the page header with the first letter converted to upper case.

```

6765 \newcommand*{\Glsxtrheadshortpl}[1]{%
6766   \protect\NoCaseChange
6767   {%
6768     \glusifattribute{#1}{headuc}{true}%

```

```

6769   {%
6770     \GLSxtrshortpl [noindex,hyper=false] {#1} []%
6771   }%
6772   {%
6773     \GLSxtrshortpl [noindex,hyper=false] {#1} []%
6774   }%
6775 }%
6776 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

6777 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
6778   \GLSxtrshortpl [noindex,hyper=false] {#1} []%
6779 }

```

\glsxtrheadtext As above but for the text value.

```

6780 \newcommand*{\glsxtrheadtext}[1]{%
6781   \protect\NoCaseChange
6782   {%
6783     \glsifattribute{#1}{headuc}{true}%
6784     {%
6785       \GLStext [noindex,hyper=false] {#1} []%
6786     }%
6787     {%
6788       \glstext [noindex,hyper=false] {#1} []%
6789     }%
6790   }%
6791 }

```

glsxtrtitletext Command to display text value in section title and table of contents.

```

6792 \newrobustcmd*{\glsxtrtitletext}[1]{%
6793   \glstext [noindex,hyper=false] {#1} []%
6794 }

```

\Glsxtrheadtext First letter converted to upper case

```

6795 \newcommand*{\Glsxtrheadtext}[1]{%
6796   \protect\NoCaseChange
6797   {%
6798     \glsifattribute{#1}{headuc}{true}%
6799     {%
6800       \GLStext [noindex,hyper=false] {#1} []%
6801     }%
6802     {%
6803       \Glstext [noindex,hyper=false] {#1} []%
6804     }%
6805   }%
6806 }

```

Glsxtrtitletext Command to display text value in section title and table of contents with the first letter changed to upper case.

```
6807 \newrobustcmd*{\Glsxtrtitletext}[1]{%
6808   \Gls{text[noindex,hyper=false]}{#1}[]%
6809 }
```

lsxtrheadplural As above but for the plural value.

```
6810 \newcommand*{\glsxtrheadplural}[1]{%
6811   \protect\NoCaseChange
6812   {%
6813     \glsifattribute{#1}{headuc}{true}%
6814     {%
6815       \GLSplural[noindex,hyper=false]{#1}[]%
6816     }%
6817     {%
6818       \glsplural[noindex,hyper=false]{#1}[]%
6819     }%
6820   }%
6821 }
```

sxtrtitleplural Command to display plural value in section title and table of contents.

```
6822 \newrobustcmd*{\glsxtrtitleplural}[1]{%
6823   \glsplural[noindex,hyper=false]{#1}[]%
6824 }
```

lsxtrheadplural Convert first letter to upper case.

```
6825 \newcommand*{\Glsxtrheadplural}[1]{%
6826   \protect\NoCaseChange
6827   {%
6828     \glsifattribute{#1}{headuc}{true}%
6829     {%
6830       \GLSplural[noindex,hyper=false]{#1}[]%
6831     }%
6832     {%
6833       \Glsplural[noindex,hyper=false]{#1}[]%
6834     }%
6835   }%
6836 }
```

sxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
6837 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
6838   \Glsplural[noindex,hyper=false]{#1}[]%
6839 }
```

glsxtrheadfirst As above but for the first value.

```
6840 \newcommand*{\glsxtrheadfirst}[1]{%
6841   \protect\NoCaseChange
```

```

6842 {%
6843   \glsifattribute{#1}{headuc}{true}%
6844   {%
6845     \GLSfirst[noindex,hyper=false]{#1}[]%
6846   }%
6847   {%
6848     \glsfirst[noindex,hyper=false]{#1}[]%
6849   }%
6850 }%
6851 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```

6852 \newrobustcmd*{\lsxtrtitlefirst}[1]{%
6853   \glsfirst[noindex,hyper=false]{#1}[]%
6854 }

```

Glsxtrheadfirst First letter converted to upper case

```

6855 \newcommand*{\Glsxtrheadfirst}[1]{%
6856   \protect\NoCaseChange
6857   {%
6858     \glsifattribute{#1}{headuc}{true}%
6859     {%
6860       \GLSfirst[noindex,hyper=false]{#1}[]%
6861     }%
6862     {%
6863       \Glsfirst[noindex,hyper=false]{#1}[]%
6864     }%
6865   }%
6866 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```

6867 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
6868   \Glsfirst[noindex,hyper=false]{#1}[]%
6869 }

```

headfirstplural As above but for the firstplural value.

```

6870 \newcommand*{\glsxtrheadfirstplural}[1]{%
6871   \protect\NoCaseChange
6872   {%
6873     \glsifattribute{#1}{headuc}{true}%
6874     {%
6875       \GLSfirstplural[noindex,hyper=false]{#1}[]%
6876     }%
6877     {%
6878       \glsfirstplural[noindex,hyper=false]{#1}[]%
6879     }%
6880   }%
6881 }

```

`\titlefirstplural` Command to display firstplural value in section title and table of contents.

```
6882 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
6883   \Glsfirstplural[noindex,hyper=false]{#1}[]%
6884 }
```

`\headfirstplural` First letter converted to upper case

```
6885 \newcommand*{\Glsxtrheadfirstplural}[1]{%
6886   \protect\NoCaseChange
6887   {%
6888     \glsifattribute{#1}{headuc}{true}%
6889     {%
6890       \GLSfirstplural[noindex,hyper=false]{#1}[]%
6891     }%
6892     {%
6893       \Glsfirstplural[noindex,hyper=false]{#1}[]%
6894     }%
6895   }%
6896 }
```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
6897 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
6898   \Glsfirstplural[noindex,hyper=false]{#1}[]%
6899 }
```

`\Glsxtrheadlong` Command used to display long form in the page header.

```
6900 \newcommand*{\Glsxtrheadlong}[1]{%
6901   \protect\NoCaseChange
6902   {%
6903     \glsifattribute{#1}{headuc}{true}%
6904     {%
6905       \GLSxtrlong[noindex,hyper=false]{#1}[]%
6906     }%
6907     {%
6908       \glsxtrlong[noindex,hyper=false]{#1}[]%
6909     }%
6910   }%
6911 }
```

`\Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
6912 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
6913   \Glsxtrlong[noindex,hyper=false]{#1}[]%
6914 }
```

`\Glsxtrheadlongpl` 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.

```
6915 \newcommand*{\Glsxtrheadlongpl}[1]{%
```



```

6916 \protect\NoCaseChange
6917 {%
6918   \glsifattribute{#1}{headuc}{true}%
6919   {%
6920     \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
6921   }%
6922   {%
6923     \glsxtrlongpl[noindex,hyper=false]{#1}[]%
6924   }%
6925 }%
6926 }

```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```

6927 \newrobustcmd*{\glsxtrtitlelongpl}[1]{%
6928   \glsxtrlongpl[noindex,hyper=false]{#1}[]%
6929 }

```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

6930 \newcommand*{\Glsxtrheadlong}[1]{%
6931   \protect\NoCaseChange
6932   {%
6933     \glsifattribute{#1}{headuc}{true}%
6934     {%
6935       \GLSxtrlong[noindex,hyper=false]{#1}[]%
6936     }%
6937     {%
6938       \Glsxtrlong[noindex,hyper=false]{#1}[]%
6939     }%
6940   }%
6941 }

```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

6942 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
6943   \Glsxtrlong[noindex,hyper=false]{#1}[]%
6944 }

```

`lsextrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```

6945 \newcommand*{\Glsxtrheadlongpl}[1]{%
6946   \protect\NoCaseChange
6947   {%
6948     \glsifattribute{#1}{headuc}{true}%
6949     {%
6950       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
6951     }%
6952     {%

```

```

6953 \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
6954 }%
6955 }%
6956 }

```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

6957 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
6958 \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
6959 }

```

`\glxtrheadfull` Command used to display full form in the page header.

```

6960 \newcommand*{\glxtrheadfull}[1]{%
6961 \protect\NoCaseChange
6962 {%
6963 \glsifattribute{#1}{headuc}{true}%
6964 {%
6965 \GLSxtrfull[noindex,hyper=false]{#1}[]%
6966 }%
6967 {%
6968 \glxtrfull[noindex,hyper=false]{#1}[]%
6969 }%
6970 }%
6971 }

```

`glxstrtitlefull` Command to display full form of abbreviation in section title and table of contents.

```

6972 \newrobustcmd*{\glxstrtitlefull}[1]{%
6973 \glxtrfull[noindex,hyper=false]{#1}[]%
6974 }

```

`lxsxtrheadfullpl` 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.

```

6975 \newcommand*{\glxtrheadfullpl}[1]{%
6976 \protect\NoCaseChange
6977 {%
6978 \glsifattribute{#1}{headuc}{true}%
6979 {%
6980 \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
6981 }%
6982 {%
6983 \glxtrfullpl[noindex,hyper=false]{#1}[]%
6984 }%
6985 }%
6986 }

```

`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents.

```

6987 \newrobustcmd*{\glxstrtitlefullpl}[1]{%

```

```

6988 \glsxtrfullpl[noindex,hyper=false]{#1}[]%
6989 }

```

\Glsxtrheadfull Command used to display full form in the page header with the first letter converted to upper case.

```

6990 \newcommand*{\Glsxtrheadfull}[1]{%
6991 \protect\NoCaseChange
6992 {%
6993 \glsifattribute{#1}{headuc}{true}%
6994 {%
6995 \Glsxtrfull[noindex,hyper=false]{#1}[]%
6996 }%
6997 {%
6998 \Glsxtrfull[noindex,hyper=false]{#1}[]%
6999 }%
7000 }%
7001 }

```

\Glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

7002 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
7003 \Glsxtrfull[noindex,hyper=false]{#1}[]%
7004 }

```

\Glsxtrheadfullpl Command used to display plural full form in the page header with the first letter converted to upper case.

```

7005 \newcommand*{\Glsxtrheadfullpl}[1]{%
7006 \protect\NoCaseChange
7007 {%
7008 \glsifattribute{#1}{headuc}{true}%
7009 {%
7010 \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
7011 }%
7012 {%
7013 \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
7014 }%
7015 }%
7016 }

```

\Glsxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

7017 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
7018 \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
7019 }

```

\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.

```

7020 \ifdef\texorpdfstring

```

```

7021 {
7022   \newcommand*{\glsfmtshort}[1]{%
7023     \texorpdfstring
7024       {\glsxtrtitleshort{#1}}%
7025       {\glsentryshort{#1}}%
7026   }
7027 }
7028 {
7029   \newcommand*{\glsfmtshort}[1]{%
7030     \glsxtrtitleshort{#1}}
7031 }

```

Similarly for the plural version.

`\glsfmtshortpl`

```

7032 \ifdef\texorpdfstring
7033 {
7034   \newcommand*{\glsfmtshortpl}[1]{%
7035     \texorpdfstring
7036       {\glsxtrtitleshortpl{#1}}%
7037       {\glsentryshortpl{#1}}%
7038   }
7039 }
7040 {
7041   \newcommand*{\glsfmtshortpl}[1]{%
7042     \glsxtrtitleshortpl{#1}}
7043 }

```

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).

```

7044 \ifdef\texorpdfstring
7045 {
7046   \newcommand*{\Glsfmtshort}[1]{%
7047     \texorpdfstring
7048       {\Glsxtrtitleshort{#1}}%
7049       {\glsentryshort{#1}}%
7050   }
7051 }
7052 {
7053   \newcommand*{\Glsfmtshort}[1]{%
7054     \Glsxtrtitleshort{#1}}
7055 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

7056 \ifdef\texorpdfstring
7057 {
7058   \newcommand*{\Glsfmtshortpl}[1]{%
7059     \texorpdfstring

```

```

7060     {\Glsxtrtitleshortpl{#1}}%
7061     {\glsentryshortpl{#1}}%
7062   }
7063 }
7064 {
7065   \newcommand*{\Glsfmtshortpl}[1]{%
7066     \Glsxtrtitleshortpl{#1}}
7067 }

```

`\glsfmttext` As above but for the text value.

```

7068 \ifdef\texorpdfstring
7069 {
7070   \newcommand*{\glsfmttext}[1]{%
7071     \texorpdfstring
7072     {\Glsxtrtitletext{#1}}%
7073     {\glsentrytext{#1}}%
7074   }
7075 }
7076 {
7077   \newcommand*{\glsfmttext}[1]{%
7078     \Glsxtrtitletext{#1}}
7079 }

```

`\Glsfmttext` First letter converted to upper case.

```

7080 \ifdef\texorpdfstring
7081 {
7082   \newcommand*{\Glsfmttext}[1]{%
7083     \texorpdfstring
7084     {\Glsxtrtitletext{#1}}%
7085     {\glsentrytext{#1}}%
7086   }
7087 }
7088 {
7089   \newcommand*{\Glsfmttext}[1]{%
7090     \Glsxtrtitletext{#1}}
7091 }

```

`\glsfmtplural` As above but for the plural value.

```

7092 \ifdef\texorpdfstring
7093 {
7094   \newcommand*{\glsfmtplural}[1]{%
7095     \texorpdfstring
7096     {\Glsxtrtitleplural{#1}}%
7097     {\glsentryplural{#1}}%
7098   }
7099 }
7100 {
7101   \newcommand*{\glsfmtplural}[1]{%
7102     \Glsxtrtitleplural{#1}}

```

7103 }

`\Glsfmtplural` First letter converted to upper case.

```
7104 \ifdef\teorpdfstring
7105 {
7106   \newcommand*\Glsfmtplural}[1]{%
7107     \teorpdfstring
7108     {\Glsxtrtitleplural{#1}}%
7109     {\glstryplural{#1}}%
7110   }
7111 }
7112 {
7113   \newcommand*\Glsfmtplural}[1]{%
7114     \Glsxtrtitleplural{#1}}
7115 }
```

`\glsfmtfirst` As above but for the first value.

```
7116 \ifdef\teorpdfstring
7117 {
7118   \newcommand*\glsfmtfirst}[1]{%
7119     \teorpdfstring
7120     {\glsxtrtitlefirst{#1}}%
7121     {\glstryfirst{#1}}%
7122   }
7123 }
7124 {
7125   \newcommand*\glsfmtfirst}[1]{%
7126     \glsxtrtitlefirst{#1}}
7127 }
```

`\Glsfmtfirst` First letter converted to upper case.

```
7128 \ifdef\teorpdfstring
7129 {
7130   \newcommand*\Glsfmtfirst}[1]{%
7131     \teorpdfstring
7132     {\Glsxtrtitlefirst{#1}}%
7133     {\glstryfirst{#1}}%
7134   }
7135 }
7136 {
7137   \newcommand*\Glsfmtfirst}[1]{%
7138     \Glsxtrtitlefirst{#1}}
7139 }
```

`\glsfmtfirstpl` As above but for the firstplural value.

```
7140 \ifdef\teorpdfstring
7141 {
7142   \newcommand*\glsfmtfirstpl}[1]{%
7143     \teorpdfstring
```

```

7144     {\glxtrtitlefirstplural{#1}}%
7145     {\glsentryfirstplural{#1}}%
7146   }
7147 }
7148 {
7149   \newcommand*{\glsfmtfirstpl}[1]{%
7150     \glxtrtitlefirstplural{#1}}
7151 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

7152 \ifdef\teorpdfstring
7153 {
7154   \newcommand*{\Glsfmtfirstpl}[1]{%
7155     \teorpdfstring
7156     {\glxtrtitlefirstplural{#1}}%
7157     {\glsentryfirstplural{#1}}%
7158   }
7159 }
7160 {
7161   \newcommand*{\Glsfmtfirstpl}[1]{%
7162     \glxtrtitlefirstplural{#1}}
7163 }

```

`\glsfmtlong` As above but for the long value.

```

7164 \ifdef\teorpdfstring
7165 {
7166   \newcommand*{\glsfmtlong}[1]{%
7167     \teorpdfstring
7168     {\glxtrtitlelong{#1}}%
7169     {\glsentrylong{#1}}%
7170   }
7171 }
7172 {
7173   \newcommand*{\glsfmtlong}[1]{%
7174     \glxtrtitlelong{#1}}
7175 }

```

`\Glsfmtlong` First letter converted to upper case.

```

7176 \ifdef\teorpdfstring
7177 {
7178   \newcommand*{\Glsfmtlong}[1]{%
7179     \teorpdfstring
7180     {\glxtrtitlelong{#1}}%
7181     {\glsentrylong{#1}}%
7182   }
7183 }
7184 {
7185   \newcommand*{\Glsfmtlong}[1]{%
7186     \glxtrtitlelong{#1}}

```

7187 }

`\glsfmtlongpl` As above but for the longplural value.

```
7188 \ifdef\txorpdfstring
7189 {
7190   \newcommand*{\glsfmtlongpl}[1]{%
7191     \txorpdfstring
7192     {\glsxtrtitlelongpl{#1}}%
7193     {\glsentrylongpl{#1}}%
7194   }
7195 }
7196 {
7197   \newcommand*{\glsfmtlongpl}[1]{%
7198     \glsxtrtitlelongpl{#1}}
7199 }
```

`\Glsfmtlongpl` First letter converted to upper case.

```
7200 \ifdef\txorpdfstring
7201 {
7202   \newcommand*{\Glsfmtlongpl}[1]{%
7203     \txorpdfstring
7204     {\Glsxtrtitlelongpl{#1}}%
7205     {\glsentrylongpl{#1}}%
7206   }
7207 }
7208 {
7209   \newcommand*{\Glsfmtlongpl}[1]{%
7210     \Glsxtrtitlelongpl{#1}}
7211 }
```

`\glsfmtfull` In-line full format.

```
7212 \ifdef\txorpdfstring
7213 {
7214   \newcommand*{\glsfmtfull}[1]{%
7215     \txorpdfstring
7216     {\glsxtrtitlefull{#1}}%
7217     {\glsxtrinlinefullformat{#1}-{}}%
7218   }
7219 }
7220 {
7221   \newcommand*{\glsfmtfull}[1]{%
7222     \glsxtrtitlefull{#1}}
7223 }
```

`\Glsfmtfull` First letter converted to upper case.

```
7224 \ifdef\txorpdfstring
7225 {
7226   \newcommand*{\Glsfmtfull}[1]{%
7227     \txorpdfstring
```



```

7228     {\Glsxtrtitlefull{#1}}%
7229     {\Glsxtrinlinefullformat{#1}-{}}%
7230   }
7231 }
7232 {
7233   \newcommand*{\Glsfmtfull}[1]{%
7234     \Glsxtrtitlefull{#1}}
7235 }

```

`\glsfmtfullpl` In-line full plural format.

```

7236 \ifdef\texorpdfstring
7237 {
7238   \newcommand*{\glsfmtfullpl}[1]{%
7239     \texorpdfstring
7240     {\Glsxtrtitlefullpl{#1}}%
7241     {\Glsxtrinlinefullplformat{#1}-{}}%
7242   }
7243 }
7244 {
7245   \newcommand*{\glsfmtfullpl}[1]{%
7246     \Glsxtrtitlefullpl{#1}}
7247 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

7248 \ifdef\texorpdfstring
7249 {
7250   \newcommand*{\Glsfmtfullpl}[1]{%
7251     \texorpdfstring
7252     {\Glsxtrtitlefullpl{#1}}%
7253     {\Glsxtrinlinefullplformat{#1}-{}}%
7254   }
7255 }
7256 {
7257   \newcommand*{\Glsfmtfullpl}[1]{%
7258     \Glsxtrtitlefullpl{#1}}
7259 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

7260 \newcommand*{\RequireGlossariesExtraLang}[1]{%
7261   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
7262 }

```

`sariesExtraLang`

```

7263 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
7264   \ProvidesFile{glossariesxtr-#1.ldf}%
7265 }

```

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.)

```

7266 \@ifpackageloaded{tracklang}
7267 {%
7268   \AnyTrackedLanguages
7269   {%
7270     \ForEachTrackedDialect{\this@dialect}{%
7271       \IfTrackedLanguageFileExists{\this@dialect}%
7272       {glossariesxtr-}% prefix
7273       {.ldf}%
7274       {%
7275         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
7276       }%
7277     }%
7278   }%
7279 }%
7280 }%
7281 {}%
7282 }
7283 {}

```

Load glossaries-extra-stylemods if required.

```

7284 \@glsxtr@redefstyles

```

and set the style:

```

7285 \@glsxtr@do@style

```

2 Style Adjustments (glossaries-extra-stylemods.sty)

This package adjusts the predefined styles so that they include the post description hook. Also, some other minor adjustments may be made to make existing styles more flexible.

2.1 Package Initialisation

First identify package:

```
7286 \NeedsTeXFormat{LaTeX2e}
7287 \ProvidesPackage{glossaries-extra-stylemods}[2017/02/07 v1.13 (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 `\@glsxtr@loadstyles`.

`sxtr@loadstyles`

```
7288 \newcommand*{\@glsxtr@loadstyles}{%
7289 \DeclareOption*{%
7290   \IfFileExists{glossary-\CurrentOption.sty}{
7291     {\eappto\@glsxtr@loadstyles{%
7292       \noexpand\RequirePackage{glossary-\CurrentOption}}}%
7293     {\PackageError{glossaries-extra-styles}%
7294       {Unknown option '\CurrentOption'}{}}
7295 }
```

Process the package options:

```
7296 \ProcessOptions
```

Load the required packages:

```
7297 \@glsxtr@loadstyles
```

Adjust the styles that the post description hook added, but only for styles that have already been defined. All the tree styles in `glossary-tree` include the post description hook, so they don't require adjustment. Similarly for `glossary-mcols` which builds on the tree styles.

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

`ewglossarystyle`

```
7298 \providecommand{\renewglossarystyle}[2]{%
7299   \ifcsundef{@glsstyle@#1}%
7300   {%
7301     \PackageError{glossaries}{Glossary style '#1' isn't already defined}{}%
```

```

7302 }%
7303 {%
7304   \csdef{@glsstyle@#1}{#2}%
7305 }%
7306 }

```

2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying.

```

7307 \ifdefined{@glsstyle@listdotted}
7308 {%
7309   \renewglossarystyle{listdotted}{%
7310     \setglossarystyle{list}%
7311     \renewcommand*{\glossentry}[2]{%
7312       \item[]\makebox[\glslistdottedwidth][l]{%
7313         \glsentryitem{##1}%
7314         \glstarget{##1}{\glossentryname{##1}}%
7315         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
7316       \glossentrydesc{##1}\glspostdescription}%
7317   \renewcommand*{\subglossentry}[3]{%
7318     \item[]\makebox[\glslistdottedwidth][l]{%
7319       \glssubentryitem{##2}%
7320       \glstarget{##2}{\glossentryname{##2}}%
7321       \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
7322     \glossentrydesc{##2}\glspostdescription}%
7323   }
7324 }
7325 {}

```

The sublistdotted style doesn't display the description for top-level entries. Sub-level entries use the listdottedstyle.

2.3 Longtable Styles

The three and four column styles require adjustment, but not the two column styles.

```

7326 \ifcsdef{@glsstyle@long3col}
7327 {%
7328   \renewglossarystyle{long3col}{%
7329     \renewenvironment{theglossary}%
7330       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
7331       {\end{longtable}}}%
7332   \renewcommand*{\glossaryheader}{}%
7333   \renewcommand*{\glsgroupheading}[1]{}%
7334   \renewcommand{\glossentry}[2]{%
7335     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7336     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline

```

```

7337 }%
7338 \renewcommand{\subglossentry}[3]{%
7339     &
7340     \glssubentryitem{##2}%
7341     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7342     ##3\tabularnewline
7343 }%
7344 \renewcommand*\{ \glsgroupskip }{%
7345     \ifglsgnoglobskip\else & &\tabularnewline\fi}%
7346 }
7347 }
7348 {}

```

Four column style:

```

7349 \ifcsdef{@glsstyle@long4col}
7350 {%
7351     \renewglossarystyle{long4col}{%
7352         \renewenvironment{theglossary}%
7353             {\begin{longtable}{llll}}%
7354             {\end{longtable}}%
7355         \renewcommand*\{ \glossaryheader }{%
7356             \renewcommand*\{ \glsgroupheading }[1]{%
7357                 \renewcommand{\glossentry}[2]{%
7358                     \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7359                     \glossentrydesc{##1}\glspostdescription &
7360                     \glossentrysymbol{##1} &
7361                     ##2\tabularnewline
7362                 }%
7363                 \renewcommand{\subglossentry}[3]{%
7364                     &
7365                     \glssubentryitem{##2}%
7366                     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7367                     \glossentrysymbol{##2} & ##3\tabularnewline
7368                 }%
7369                 \renewcommand*\{ \glsgroupskip }{%
7370                     \ifglsgnoglobskip\else & &\tabularnewline\fi}%
7371             }
7372         }
7373     }

```

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, but not the two column styles.

```

7374 \ifcsdef{@glsstyle@longragged3col}
7375 {%
7376     \renewglossarystyle{longragged3col}{%

```

```

7377 \renewenvironment{theglossary}%
7378     {\begin{longtable}[l>{\raggedright}p{\glsgdescwidth}%
7379      >{\raggedright}p{\glspagelistwidth}}}%
7380     {\end{longtable}}}%
7381 \renewcommand*{\glossaryheader}{}%
7382 \renewcommand*{\glsgroupheading}[1]{}%
7383 \renewcommand{\glossentry}[2]{%
7384     \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7385     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
7386 }%
7387 \renewcommand{\subglossentry}[3]{%
7388     &
7389     \glssubentryitem{##2}%
7390     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7391     ##3\tabularnewline
7392 }%
7393 \renewcommand*{\glsgroupskip}{%
7394     \ifglsgnogroupskip\else & \tabularnewline\fi}%
7395 }
7396 }
7397 {}

```

Four column style:

```

7398 \ifcsdef{@glstyle@altlongragged4col}
7399 {%
7400     \renewglossarystyle{altlongragged4col}{%
7401         \renewenvironment{theglossary}%
7402             {\begin{longtable}[l>{\raggedright}p{\glsgdescwidth}l%
7403              >{\raggedright}p{\glspagelistwidth}}}%
7404             {\end{longtable}}}%
7405         \renewcommand*{\glossaryheader}{}%
7406         \renewcommand*{\glsgroupheading}[1]{}%
7407         \renewcommand{\glossentry}[2]{%
7408             \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7409             \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
7410             ##2\tabularnewline
7411         }%
7412         \renewcommand{\subglossentry}[3]{%
7413             &
7414             \glssubentryitem{##2}%
7415             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7416             \glossentrysymbol{##2} & ##3\tabularnewline
7417         }%
7418         \renewcommand*{\glsgroupskip}{%
7419             \ifglsgnogroupskip\else & & \tabularnewline\fi}%
7420     }
7421 }
7422 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment, but not the two column styles.

```
7423 \ifcsdef{@glsstyle@super3col}
7424 {%
7425   \renewglossarystyle{super3col}{%
7426     \renewenvironment{theglossary}%
7427       {\tablehead{}}\tabletail{}}%
7428     \begin{supertabular}{\lp{@glsdescwidth}p{@glspagelistwidth}}}%
7429     {\end{supertabular}}}%
7430   \renewcommand*{\glossaryheader}{}%
7431   \renewcommand*{\glsgroupheading}[1]{}%
7432   \renewcommand{\glossentry}[2]{%
7433     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7434     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
7435   }%
7436   \renewcommand{\subglossentry}[3]{%
7437     &
7438     \glssubentryitem{##2}%
7439     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7440     ##3\tabularnewline
7441   }%
7442   \renewcommand*{\glsgroupskip}{%
7443     \ifglsnogroupskip\else & &\tabularnewline\fi}%
7444 }
7445 }
7446 {}
```

Four column styles:

```
7447 \ifcsdef{@glsstyle@super4col}
7448 {%
7449   \renewglossarystyle{super4col}{%
7450     \renewenvironment{theglossary}%
7451       {\tablehead{}}\tabletail{}}%
7452     \begin{supertabular}{\llll}}}%
7453     \end{supertabular}}}%
7454   \renewcommand*{\glossaryheader}{}%
7455   \renewcommand*{\glsgroupheading}[1]{}%
7456   \renewcommand{\glossentry}[2]{%
7457     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7458     \glossentrydesc{##1}\glspostdescription &
7459     \glossentrysymbol{##1} & ##2\tabularnewline
7460   }%
7461   \renewcommand{\subglossentry}[3]{%
7462     &
7463     \glssubentryitem{##2}%
7464     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7465     \glossentrysymbol{##2} & ##3\tabularnewline
7466   }%
7467   \renewcommand*{\glsgroupskip}{%

```

```

7468     \ifglsnogroupskip\else & & \tabularnewline\fi}%
7469 }
7470 }
7471 {}

```

2.6 Super Ragged Styles

The three and four column styles require adjustment, but not the two column styles.

```

7472 \ifcsdef{@glsstyle@superragged3col}
7473 {%
7474   \renewglossarystyle{superragged3col}{%
7475     \renewenvironment{theglossary}%
7476       {\tablehead{}\tabletail}%
7477       \begin{supertabular}{1>\raggedright}p{\glsdescwidth}%
7478         >\raggedright}p{\glspagelistwidth}}}%
7479     {\end{supertabular}}}%
7480   \renewcommand*{\glossaryheader}{}%
7481   \renewcommand*{\glsgroupheading}[1]{}%
7482   \renewcommand{\glossentry}[2]{%
7483     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7484     \glossentrydesc{##1}\glspostdescription &
7485     ##2\tabularnewline
7486   }%
7487   \renewcommand{\subglossentry}[3]{%
7488     &
7489     \glssubentryitem{##2}%
7490     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7491     ##3\tabularnewline
7492   }%
7493   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
7494     &\tabularnewline\fi}%
7495 }
7496 }
7497 {}

```

Four columns:

```

7498 \ifcsdef{@glsstyle@altsuperragged4col}
7499 {%
7500   \renewglossarystyle{altsuperragged4col}{%
7501     \renewenvironment{theglossary}%
7502       {\tablehead{}\tabletail}%
7503       \begin{supertabular}{1>\raggedright}p{\glsdescwidth}l%
7504         >\raggedright}p{\glspagelistwidth}}}%
7505     {\end{supertabular}}}%
7506   \renewcommand*{\glossaryheader}{}%
7507   \renewcommand{\glossentry}[2]{%
7508     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7509     \glossentrydesc{##1}\glspostdescription &
7510     \glossentrysymbol{##1} & ##2\tabularnewline

```



```

7511 }%
7512 \renewcommand{\subglossentry}[3]{%
7513     &
7514     \glssubentryitem{##2}%
7515     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7516     \glossentrysymbol{##2} & ##3\tabularnewline
7517 }%
7518 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & &
7519     &\tabularnewline\fi}%
7520 }
7521 }
7522 {}

```

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`.

```

7523 \ifdef{\@glsstyle@inline}
7524 {%
7525     \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
7526     Just use \glxtrpostdescription instead of \glspostdescription.
7527     \renewcommand*{\glsinlinedescformat}[3]{%
7528         \space#1\glxtrpostdescription}
7529     \renewcommand*{\glsinlinesubdescformat}[3]{%
7530         #1\glxtrpostdescription}
7531 }

```

2.8 Tree Styles

The `alttree` style is redefined to make it easier to made minor adjustments.

```

7532 \ifdef{\@glsstyle@alttree}
7533 {%

```

Only redefine this style if it's already been defined.

mbolDescLocation

```
\glxtraltrtreeSymbolDescLocation{\label}{\location list}
```

Layout the symbol, description and location for top-level entries.

```

7534 \newcommand{\glxtraltrtreeSymbolDescLocation}[2]{%
7535     {%
7536         \let\par\glxtrAltTreePar

```

```

7537     \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
7538     \glossentrydesc{#1}\glspostdescription \space #2\par
7539 }%
7540 }

```

`\trAltTreeIndent` Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

7541 \newlength\glxstrAltTreeIndent

```

`\lsxtrAltTreePar` Multi-paragraph descriptions need to keep the hanging indent.

```

7542 \newcommand{\glxstrAltTreePar}{%
7543   @@par
7544   \glxstrAltTreeSetHangIndent
7545   \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
7546 }

```

`\symbolDescLocation` `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

7547 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
7548   \glxstralttreeSymbolDescLocation{#2}{#3}%
7549 }

```

`\trtreeindent` 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.

```

7550 \newlength\glxstrtreeindent

```

`\glxstralttreeInit` User-level initialisation for the alttree style.

```

7551 \newcommand*{\glxstralttreeInit}{%
7552   \settowidth{\glxstrtreeindent}{\glstreenamfmt{\glsgetwidestname\space}}%
7553   \glxstrAltTreeIndent=\parindent
7554 }

```

`\eglissetwidest` The original `\glissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

7555 \newcommand*{\eglissetwidest}[2][0]{%
7556   \protected@csedef{@glswidestname\romannumeral#1}{#2}%
7557 }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```

7558 \newcommand*{\xglissetwidest}[2][0]{%
7559   \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
7560 }

```

`\lsgetwidestname` Provide a user-level macro to obtain the widest top-level name.

```

7561 \newcommand*{\lsgetwidestname}{\@glswidestname}

```

`\getwidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```

7562 \newcommand*\glsgetwidestsubname}[1]{%
7563   \ifcsundef{@glswidestname\romannumeral#1}%
7564     {\@glswidestname}%
7565     {\csuse{@glswidestname\romannumeral#1}}%
7566 }

```

`\findTopLevelName` CamelCase is easier for long command names. Provide a CamelCase synonym of `\glsfindwidesttoplevelname`

```

7567 \let\glsFindWidestTopLevelName\glsfindwidesttoplevelname

```

`\usedTopLevelName` Like `\glsfindwidesttoplevelname` 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.

```

7568 \newrobustcmd*\glsFindWidestUsedTopLevelName}[1][\@glo@types]{%
7569   \dimen@=0pt\relax
7570   \gls@tmplen=0pt\relax
7571   \foralllglossaries[#1]{\@gls@type}%
7572   {%
7573     \forglsentries[\@gls@type]{\@glo@label}%
7574     {%
7575       \ifglsused{\@glo@label}%
7576       {%
7577         \ifglshasparent{\@glo@label}%
7578         {}%
7579         {%
7580           \settowidth{\dimen@}%
7581             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
7582           \ifdim\dimen@>\gls@tmplen
7583             \gls@tmplen=\dimen@
7584             \eglssetwidest{\glsentryname{\@glo@label}}%
7585           \fi
7586         }%
7587       }%
7588     }%
7589   }%
7590 }%
7591 }

```

`\destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

7592 \newrobustcmd*\glsFindWidestUsedAnyName}[1][\@glo@types]{%
7593   \dimen@=0pt\relax
7594   \gls@tmplen=0pt\relax
7595   \foralllglossaries[#1]{\@gls@type}%
7596   {%
7597     \forglsentries[\@gls@type]{\@glo@label}%
7598     {%

```

```

7599     \ifglsused{\@glo@label}%
7600     {%
7601         \settowidth{\dimen@}%
7602             {\glstreenamefmt{\glstentryname{\@glo@label}}}%
7603         \ifdim\dimen@>\gls@tmplen
7604             \gls@tmplen=\dimen@
7605             \eglssetwidest{\glstentryname{\@glo@label}}%
7606         \fi
7607     }%
7608     {%
7609     }%
7610 }%
7611 }

```

FindWidestAnyName Like the above but doesn't check if the entry has been used.

```

7612 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
7613     \dimen@=0pt\relax
7614     \gls@tmplen=0pt\relax
7615     \forallglossaries[#1]{\@gls@type}%
7616     {%
7617         \forallglsentries[\@gls@type]{\@glo@label}%
7618         {%
7619             \settowidth{\dimen@}%
7620                 {\glstreenamefmt{\glstentryname{\@glo@label}}}%
7621             \ifdim\dimen@>\gls@tmplen
7622                 \gls@tmplen=\dimen@
7623                 \eglssetwidest{\glstentryname{\@glo@label}}%
7624             \fi
7625         }%
7626     }%
7627 }

```

FindUsedLevelTwo This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

7628 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
7629     \dimen@=0pt\relax
7630     \dimen@i=0pt\relax
7631     \dimen@ii=0pt\relax
7632     \forallglossaries[#1]{\@gls@type}%
7633     {%
7634         \forallglsentries[\@gls@type]{\@glo@label}%
7635         {%
7636             \ifglsused{\@glo@label}%
7637             {%
7638                 \ifglshasparent{\@glo@label}%
7639                 {%
7640                     \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
7641                     \ifglshasparent{\@glo@parent}%
7642                 }%

```

```

7643      \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
7644      \ifglshasparent{\@glo@parent}%
7645      {%
7646      {%
7647          \settowidth{\gls@tmplen}%
7648              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7649          \ifdim\gls@tmplen>\dimen@ii
7650              \dimen@ii=\gls@tmplen
7651              \eglssetwidest[2]{\glsentryname{\@glo@label}}%
7652          \fi
7653      }%
7654      }%
7655      {%
7656          \settowidth{\gls@tmplen}%
7657              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7658          \ifdim\gls@tmplen>\dimen@i
7659              \dimen@i=\gls@tmplen
7660              \eglssetwidest[1]{\glsentryname{\@glo@label}}%
7661          \fi
7662      }%
7663      }%
7664      {%
7665          \settowidth{\gls@tmplen}%
7666              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7667          \ifdim\gls@tmplen>\dimen@
7668              \dimen@=\gls@tmplen
7669              \eglssetwidest{\glsentryname{\@glo@label}}%
7670          \fi
7671      }%
7672      }%
7673      {}%
7674      }%
7675      }%
7676  }

```

dWidestLevelTwo This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

7677  \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
7678      \dimen@=0pt\relax
7679      \dimen@i=0pt\relax
7680      \dimen@ii=0pt\relax
7681      \foralllglossaries[#1]{\@gls@type}%
7682      {%
7683          \forglsentries[\@gls@type]{\@glo@label}%
7684          {%
7685              \ifglshasparent{\@glo@label}%
7686              {%
7687                  \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
7688                  \ifglshasparent{\@glo@parent}%
7689                  {%

```

```

7690 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
7691 \ifglshasparent{\@glo@parent}%
7692 {}%
7693 {%
7694 \settowidth{\gls@tmplen}%
7695 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7696 \ifdim\gls@tmplen>\dimen@ii
7697 \dimen@ii=\gls@tmplen
7698 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
7699 \fi
7700 }%
7701 }%
7702 {%
7703 \settowidth{\gls@tmplen}%
7704 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7705 \ifdim\gls@tmplen>\dimen@i
7706 \dimen@i=\gls@tmplen
7707 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
7708 \fi
7709 }%
7710 }%
7711 {%
7712 \settowidth{\gls@tmplen}%
7713 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7714 \ifdim\gls@tmplen>\dimen@
7715 \dimen@=\gls@tmplen
7716 \eglssetwidest{\glsentryname{\@glo@label}}%
7717 \fi
7718 }%
7719 }%
7720 }%
7721 }

```

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.

```

7722 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
7723 \dimen@=0pt\relax
7724 \gls@tmplen=0pt\relax
7725 #2=0pt\relax
7726 \forallglossaries[#1]{\@gls@type}%
7727 {%
7728 \forglsentries[\@gls@type]{\@glo@label}%
7729 {%
7730 \ifglused{\@glo@label}%
7731 {%
7732 \settowidth{\dimen@}%
7733 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7734 \ifdim\dimen@>\gls@tmplen
7735 \gls@tmplen=\dimen@

```

```

7736         \eglssetwidest{\glsentryname{\@glo@label}}%
7737     \fi
7738     \settowidth{\dimen@}%
7739     {\glsentrysymbol{\@glo@label}}%
7740     \ifdim\dimen@>#2\relax
7741         #2=\dimen@
7742     \fi
7743 }%
7744 {}%
7745 }%
7746 }%
7747 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

7748 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
7749     \dimen@=0pt\relax
7750     \gls@tmplen=0pt\relax
7751     #2=0pt\relax
7752     \forallglossaries[#1]{\@gls@type}%
7753     {%
7754         \forglsentries[\@gls@type]{\@glo@label}%
7755         {%
7756             \settowidth{\dimen@}%
7757             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
7758             \ifdim\dimen@>\gls@tmplen
7759                 \gls@tmplen=\dimen@
7760                 \eglssetwidest{\glsentryname{\@glo@label}}%
7761             \fi
7762             \settowidth{\dimen@}%
7763             {\glsentrysymbol{\@glo@label}}%
7764             \ifdim\dimen@>#2\relax
7765                 #2=\dimen@
7766             \fi
7767         }%
7768     }%
7769 }

```

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.

```

7770 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
7771     \dimen@=0pt\relax
7772     \gls@tmplen=0pt\relax
7773     #2=0pt\relax
7774     #3=0pt\relax
7775     \forallglossaries[#1]{\@gls@type}%
7776     {%
7777         \forglsentries[\@gls@type]{\@glo@label}%

```

```

7778     {%
7779         \ifglsused{\@glo@label}%
7780     {%
7781         \settowidth{\dimen@}%
7782             {\glstreenamefmt{\glentryname{\@glo@label}}}%
7783         \ifdim\dimen@>\gls@tmplen
7784             \gls@tmplen=\dimen@
7785             \eglssetwidest{\glentryname{\@glo@label}}%
7786         \fi
7787         \settowidth{\dimen@}%
7788             {\glentrysymbol{\@glo@label}}%
7789         \ifdim\dimen@>#2\relax
7790             #2=\dimen@
7791         \fi
7792         \settowidth{\dimen@}%
7793             {\GlsXtrFormatLocationList{\glentrynumberlist{\@glo@label}}}%
7794         \ifdim\dimen@>#3\relax
7795             #3=\dimen@
7796         \fi
7797     }%
7798 }%
7799 }%
7800 }%
7801 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

7802 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
7803     \dimen@=0pt\relax
7804     \gls@tmplen=0pt\relax
7805     #2=0pt\relax
7806     #3=0pt\relax
7807     \forallglossaries[#1]{\@gls@type}%
7808     {%
7809         \forglsentries[\@gls@type]{\@glo@label}%
7810     {%
7811         \settowidth{\dimen@}%
7812             {\glstreenamefmt{\glentryname{\@glo@label}}}%
7813         \ifdim\dimen@>\gls@tmplen
7814             \gls@tmplen=\dimen@
7815             \eglssetwidest{\glentryname{\@glo@label}}%
7816         \fi
7817         \settowidth{\dimen@}%
7818             {\glentrysymbol{\@glo@label}}%
7819         \ifdim\dimen@>#2\relax
7820             #2=\dimen@
7821         \fi
7822         \settowidth{\dimen@}%
7823             {\GlsXtrFormatLocationList{\glentrynumberlist{\@glo@label}}}%
7824         \ifdim\dimen@>#3\relax

```



```

7825         #3=\dimen@
7826     \fi
7827 }%
7828 }%
7829 }

```

AnyNameLocation Like the `\glsFindWidestUsedAnyNameSymbolLocation` but doesn't measure the symbol. The length of the widest location list is stored in the second argument, which should be a length register.

```

7830 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
7831     \dimen@=0pt\relax
7832     \gls@tmplen=0pt\relax
7833     #2=0pt\relax
7834     \forallglossaries[#1]{\@gls@type}%
7835     {%
7836         \forglstentries[\@gls@type]{\@glo@label}%
7837         {%
7838             \ifglstused{\@glo@label}%
7839             {%
7840                 \settowidth{\dimen@}%
7841                 {\glstreenamfmt{\glstentryname{\@glo@label}}}%
7842                 \ifdim\dimen@>\gls@tmplen
7843                 \gls@tmplen=\dimen@
7844                 \eglissetwidest{\glstentryname{\@glo@label}}%
7845             \fi
7846             \settowidth{\dimen@}%
7847             {\GlsXtrFormatLocationList{\glstentrynumberlist{\@glo@label}}}%
7848             \ifdim\dimen@>#2\relax
7849                 #2=\dimen@
7850             \fi
7851         }%
7852     }%
7853 }%
7854 }%
7855 }

```

AnyNameLocation Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

7856 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
7857     \dimen@=0pt\relax
7858     \gls@tmplen=0pt\relax
7859     #2=0pt\relax
7860     \forallglossaries[#1]{\@gls@type}%
7861     {%
7862         \forglstentries[\@gls@type]{\@glo@label}%
7863         {%
7864             \settowidth{\dimen@}%
7865             {\glstreenamfmt{\glstentryname{\@glo@label}}}%
7866             \ifdim\dimen@>\gls@tmplen
7867                 \gls@tmplen=\dimen@

```

```

7868         \eglssetwidest{\glstryname{\@glo@label}}}%
7869     \fi
7870     \settowidth{\dimen@}%
7871         {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
7872     \ifdim\dimen@>#2\relax
7873         #2=\dimen@
7874     \fi
7875 }%
7876 }%
7877 }

```

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`.

```

7878 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
7879     \glstreeindent=\glsxtrtreetopindent\relax
7880 }

```

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.

```

7881 \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
7882     \ifcsundef{@glswidestname\romannumeral#1}%
7883     {%
7884         \settowidth{#3}{\glstreenamfmt{\@glswidestname\space}}%
7885     }%
7886     {%
7887         \settowidth{#3}{\glstreenamfmt{%
7888             \csname @glswidestname\romannumeral#1\endcsname\space}}%
7889     }%
7890 }

```

treeSetHangIndent Set `\hangindent` for top-level entries:

```

7891 \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

treeSetSubHangIndent Set `\hangindent` for sub-entries:

```

7892 \newcommand*{\glsxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine `alttree`:

```

7893 \renewglossarystyle{alttree}{%
7894     \renewenvironment{theglossary}%
7895     {%
7896         \glsxtralttreeInit

```

```

7897     \def\@gls@prevlevel{-1}%
7898     \mbox{}\par}%
7899     {\par}%
7900 \renewcommand*\glossaryheader{}%
7901 \renewcommand*\glsgroupheading[1]{}%
7902 \renewcommand\glossentry[2]{%
7903     \ifnum\@gls@prevlevel=0\relax
7904     \else
7905         \glstrComputeTreeIndent{##1}%
7906     \fi
7907     \parindent\glstreeindent
7908     \glstrAltTreeSetHangIndent
7909     \makebox[Opt][r]%
7910     {%
7911         \glstreenamebox{\glstreeindent}%
7912         {%
7913             \glstryitem{##1}%
7914             \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
7915         }%
7916     }%
7917     \glstralttreeSymbolDescLocation{##1}{##2}%
7918     \def\@gls@prevlevel{0}%
7919 }
7920 \renewcommand\subglossentry[3]{%
7921     \ifnum##1=1\relax
7922         \glssubentryitem{##2}%
7923     \fi
7924     \ifnum\@gls@prevlevel=##1\relax
7925     \else
7926         \glstrComputeTreeSubIndent{##1}{##2}{\@gls@tmplen}%
7927         \ifnum\@gls@prevlevel<##1\relax
7928             \setlength\glstreeindent\@gls@tmplen
7929             \addtolength\glstreeindent\parindent
7930             \parindent\glstreeindent
7931         \else
7932             \ifnum\@gls@prevlevel=0\relax
7933                 \glstrComputeTreeIndent{##2}%
7934             \else
7935                 \glstrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
7936             \fi
7937             \addtolength\parindent{-\glstreeindent}%
7938             \setlength\glstreeindent\parindent
7939         \fi
7940     \fi
7941     \glstrAltTreeSetSubHangIndent{##1}%
7942     \makebox[Opt][r]{\glstreenamebox{\@gls@tmplen}{%
7943         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
7944     \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
7945     \def\@gls@prevlevel{##1}%

```

```

7946     }%
7947     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7948   }
7949 }%
7950 {%

```

Assume the style isn't required if it hasn't already been defined.

```

7951 }

```

Reset the default style

```

7952 \ifx\@glossary@default@style\relax
7953 \else
7954   \setglossarystyle{\@glxtr@current@style}
7955 \fi

```

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	47
General: Initial experimental release	4	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	47
\Glsfmtshort: new	204	\@Glstext@: added redefinition	43
\glsfmtshort: new	203	\@Glsuseri@: added redefinition	48
\Glsfmtshortpl: new	204	\@Glsuserii@: added redefinition	48
\glsfmtshortpl: new	204	\@Glsuseriii@: added redefinition	48
short: switched inline full form to short		\@Glsuseriv@: added redefinition	49
(long)	166	\@Glsuserv@: added redefinition	49
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	49
\@ACRlong: added redefinition	52	\@acrlong: added redefinition	52
\@ACRlongpl: added redefinition	53	\@acrlongpl: added redefinition	53
\@ACRshort: added redefinition	50	\@acrshort: added redefinition	50
\@ACRshortpl: added redefinition	51	\@acrshortpl: added redefinition	51
\@Acrlong: added redefinition	52	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	53	argument	39
\@Acrshort: added redefinition	50	\@glsdescplural@: added redefinition .	46
\@Acrshortpl: added redefinition	51	\@glsfirst@: added redefinition	43
\@GLSdesc@: added redefinition	46	\@glsfirstplural@: added redefinition	44
\@GLSdescplural@: added redefinition .	46	\@glsplural@: added redefinition	44
\@GLSfirst@: added redefinition	44	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	45	redefinition	47
\@GLSname@: added redefinition	45	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	44	new	98
\@GLSSymbol@: added redefinition	47	\@glsxtr@field@linkdefs: new	42
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	139
redefinition	47	\@print@glossary: added redefinition .	94
\@GLStext@: added redefinition	43	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	48	\abbrvdefaultfont	143
\@GLSuserii@: added redefinition	48	\glsaccessdesc: new	109
\@GLSuseriii@: added redefinition	49	\glsaccessdescplural: new	109
\@GLSuseriv@: added redefinition	49	\glsaccessfirst: new	106
\@GLSuserv@: added redefinition	49	\glsaccessfirstplural: new	107
\@GLSuservi@: added redefinition	49	\Glsaccesslong: new	111
\@GLSdesc@: added redefinition	46	\glsaccesslong: new	111
\@GLSdescplural@: added redefinition .	46	\glsaccessname: new	105
\@Glsfirst@: added redefinition	43	\glsaccessplural: new	106
\@Glsfirstplural@: added redefinition	45	\Glsaccessshort: new	110
\@Glsname@: added redefinition	45	\glsaccessshort: new	110
\@Glsplural@: added redefinition	44	\Glsaccessshortpl: new	111

\glsaccesssshortpl: new	110	\@cGLSpl: new	76
\glsaccessssymbol: new	108	\@cGLSpl@: new	76
\glsaccessssymbolplural: new	108	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	105	new	71
\glstentryfmt: added check for short ..	39	\cGLS: new	76
\glslongpltok: new	139	\cGLSformat: new	76
\glsshortpltok: new	138	\cGLSpl: new	76
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	76
name in \setkeys	139	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	11
for plural	135	\glsenableentrycount: new	71
\GLSxtrlongpl: new	152	\glsfirstabbrvdefaultfont: new ..	142
\Glsxtrlongpl: new	151	\glsfirstlongdefaultfont: new ...	143
\glsxtrlongpl: new	151	\Glsfmtfirst: new	206
\glsxtrNoGlossaryWarning: new	14	\glsfmtfirst: new	206
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	207
new	135	\glsfmtfirstpl: new	206
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	206
new	135	\glsfmtplural: new	205
\glsxtrpostlinkendsentence: new ..	135	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	150	\Glsxtrtitleshort	204
\Glsxtrshortpl: new	150	renamed from \glstentryfmtshort ..	204
\glsxtrshortpl: new	149	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort	203
\glslabeltok	161	renamed from \glstentryfmtshort ..	203
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	159	\Glsxtrtitleshortpl	204
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl	204
redefinition of \acronymtype	11	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	204
\Glsxtrshort	204	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl	204
\glsxtrshort	203	\Glsfmttext: new	205
\Glsfmtshortpl: changed to use		\glsfmttext: new	205
\glsxtrshortpl	204	\glshasattribute: new	116
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	116
\glsxtrshortpl	204	\GlsXtrEnableEntryCounting: new ..	71
\glsxtrifemptyglossary: new	17	\glsxtrifcounttrigger: new	73
\glsxtrnewnumber: added extra		\glsxtrscfont: new	170
argument	120	\glsxtrscsuffix: new	171
\glsxtrnewsymbol: added extra		\glsxtrsmfont: new	174
argument	119	\glsxtrsmsuffix: new	175
\MakeAcronymsAbbreviations: set the		short-em: new	181
default type to \acronymtype	84	short-em-desc: new	182
\newterm: fixed name argument	119	short-em-footnote: new	183
0.5 (2015-12-07)		short-em-long: new	180
\@cGLS: new	76	short-em-long-desc: new	180
\@cGLS@: new	76	short-em-postfootnote: new	184

short-sc-footnote: new	174	\Glsxtrheadtext: now uses headuc	
short-sc-postfootnote: new	174	attribute	197
short-sm: new	176	\glxtrheadtext: now uses headuc	
short-sm-desc: new	176	attribute	197
short-sm-footnote: new	177	short-long: switch off regular attribute	
short-sm-long: new	175	if set	160
short-sm-long-desc: new	175	short-long-desc: switch off regular	
short-sm-postfootnote: new	177	attribute if set	161
long-noshort-em: new	182	long-short: switch off regular attribute	
long-noshort-em-desc: new	183	if set	158
long-noshort-sm: new	176	long-short-desc: switch off regular	
long-noshort-sm-desc: new	177	attribute if set	160
long-short-em: new	178	footnote: switch off regular attribute if	
long-short-em-desc: new	178	set	162
long-short-sm: new	175	postfootnote: switch off regular	
long-short-sm-desc: new	175	attribute if set	164
0.5.1 (2015-12-02)		0.5.2 (2015-12-08)	
\Glsaccessstext: new	105	\@GLSdesc@: added accessibility support	46
0.5.1 (2015-12-07)		\@GLSdescplural@: added accessibility	
\@glxtr@doaccsupp: new	14	support	46
General: removed \ifglxtruseuhead	195	\@GLSfirst@: added accessibility	
\Glsaccessdesc: new	109	support	44
\Glsaccessdescplural: new	109	\@GLSfirstplural@: added accessibility	
\Glsaccessfirst: new	107	support	45
\Glsaccessfirstplural: new	107	\@GLSname@: added accessibility support	45
\Glsaccessname: new	105	\@GLSplural@: added accessibility	
\Glsaccessplural: new	106	support	44
\Glsaccesssymbol: new	108	\@GLSsymbol@: added accessibility	
\Glsaccesssymbolplural: new	108	support	47
\Glsxtrheadfirst: now uses headuc		\@GLSsymbolplural@: added	
attribute	199	accessibility support	47
\glxtrheadfirst: now uses headuc		\@GLStext@: added accessibility support	43
attribute	198	\@GLSdesc@: added accessibility support	46
\Glsxtrheadfirstplural: now uses		\@GLSdescplural@: added accessibility	
headuc attribute	200	support	46
\glxtrheadfirstplural: now uses		\@GLSfirst@: added accessibility	
headuc attribute	199	support	43
\Glsxtrheadplural: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	198	support	45
\glxtrheadplural: now uses headuc		\@GLSname@: add accessibility support ..	45
attribute	198	\@GLSplural@: added accessibility	
\Glsxtrheadshort: now uses headuc		support	44
attribute	196	\@GLSsymbol@: added accessibility	
\glxtrheadshort: now uses headuc		support	47
attribute	195	\@GLSsymbolplural@: added	
\Glsxtrheadshortpl: now uses headuc		accessibility support	47
attribute	196	\@GLStext@: added accessibility support	43
\glxtrheadshortpl: now uses headuc		\@GLSdesc@: added accessibility support	46
attribute	196		

\@glsdescplural@: added accessibility support	46	\glsxtrnewabbrevpresetkeyhook: new	141
\@glsfirst@: added accessibility support	43	\glsxtrtagfont: new	133
\@glsfirstplural@: added accessibility support	44	\KV@printgloss@nonumberlist: added	38
\@glsname@: added accessibility support	45	\mfu@checkword@do: added	132
\@glsplural@: added accessibility support	44	\setabbreviationstyle: added check for post-definition style switch	155
\@glsymbol@: added accessibility support	47	0.5.3 (2015-12-09)	
\@glsymbolplural@: added accessibility support	47	\@glsxtr@autoindex@at: new	129
\@glstext@: added accessibility support	42	\@glsxtr@autoindex@encap: new ...	129
\@glsxtr@activate@initialtagging: new	133	\@glsxtr@autoindex@esc: new	130
\@glsxtr@do@titlecaps@warn: new	133	\@glsxtr@autoindex@level: new ...	130
\@glsxtr@tag: new	133	\@glsxtr@autoindex@setname: new	128
General: fixed typo in glossaries-accsupp and tidied up code to use just one		\@glsxtr@doabbreviationsdef: new	11
\@ifpackageloaded	105	General: removed	
removed \glsxtrabbrvfmt	153	\GlsXtrNoGlsWarningNoAutoMakeMain	97
\glossaryentrynumbers: added	36	\glsdescwidth: added	36
\Glossentrydesc: added	131	\glspagelistwidth: added	36
\Glossentryname: added	125	\glsxtrdoautoindexname: new	127
\Glossentrysymbol: added	131	\glsxtrpostnamehook: new	126
\glossentrysymbol: added	131	\if@glsxtr@format@override: new	127
\GLSaccessdesc: new	109, 114	\ProvidesGlossariesExtraLang: new	209
\GLSaccessdescplural: new ...	110, 114	\RequireGlossariesExtraLang: new	209
\GLSaccessfirst: new	107, 113	0.5.4 (2015-12-15)	
\GLSaccessfirstplural: new ..	107, 113	\@newglossaryentry@defunitcounters: new	77
\GLSaccesslong: new	111, 115	\@GLSxtr@p@acrlong@: new	64
\GLSaccesslongpl: new	112, 115	\@GLSxtr@p@acrlongpl@: new	65
\Glsaccesslongpl: new	112	\@GLSxtr@p@acrshort@: new	64
\glsaccesslongpl: new	111	\@GLSxtr@p@acrshortpl@: new	64
\GLSaccessname: new	105, 112	\@GLSxtr@p@long@: new	63
\GLSaccessplural: new	106, 113	\@GLSxtr@p@longpl@: new	64
\GLSaccessshort: new	110, 114	\@GLSxtr@p@plural@: new	62
\GLSaccessshortpl: new	111, 114	\@GLSxtr@p@short@: new	63
\GLSaccesssymbol: new	108, 113	\@GLSxtr@p@shortpl@: new	63
\GLSaccesssymbolplural: new	108, 113	\@GLSxtr@p@text@: new	62
\GLSaccessstext: new	106, 112	\@GlsXtrEnableOnTheFly: new	32
\glsentryfmt: moved		\@Glsxtr: new	33
\glssetabbrvfmt from		\@Glsxtr@p@acrlong@: new	64
\glsxtrabbrvfmt to here	39	\@Glsxtr@p@acrlongpl@: new	64
\GlsXtrEnableInitialTagging: new	132	\@Glsxtr@p@acrshort@: new	64
\glsxtrfieldtitlecase: new	121	\@Glsxtr@p@acrshortpl@: new	64
\GlsXtrFormatLocationList: new ...	37	\@Glsxtr@p@long@: new	63
		\@Glsxtr@p@longpl@: new	64
		\@Glsxtr@p@plural@: new	62
		\@Glsxtr@p@short@: new	63
		\@Glsxtr@p@shortpl@: new	63
		\@Glsxtr@p@text@: new	62

\@Glsxtrpl: new	34	\glxtr: new	32
\@alt@glshyp@opt: new	59	\glxtrcat: new	32
\@glscalt@hyp@opt: new	59	\glxtrdowrglossaryhook: new	59
\@glscalt@hyp@opt@char: new	59	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	59	new	82
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new	31
new	78	\Glsxtrpl: new	33
\@glsc@local@increment@currunitcount:		\glxtrpl: new	33
new	78	\glxtrpostlocalreset: new	71
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new	70
new	57	\glxtrpostreset: new	70
\@glxtr: new	32	\glxtrpostunset: new	70
\@glxtr@addunitcounter: new	77	\glxtrprotectlinks: new	61
\@glxtr@currunitcount: new	79	\GlsXtrSetAltModifier: new	60
\@glxtr@ifunitcounter: new	77	\GlsXtrSetDefaultGlsOpts: new	58
\@glxtr@p@acrlong@: new	64	\glxtrstarflywarn: new	32
\@glxtr@p@acrlongpl@: new	64	\GlsXtrWarning: new	34
\@glxtr@p@acrshort@: new	64	\MakeAcronymsAbbreviations: now	
\@glxtr@p@acrshortpl@: new	64	disables \setacronymstyle	84
\@glxtr@p@long@: new	63	1.0 (2016-01-24)	
\@glxtr@p@longpl@: new	64	\@glxtr@autoindexcrossrefs: new .	10
\@glxtr@p@plural@: new	62	\@glxtr@idx@displaynumberlist:	
\@glxtr@p@short@: new	62	new	91
\@glxtr@p@shortpl@: new	63	\@glxtr@idx@entrynumberlist: new	92
\@glxtr@p@text@: new	62	\@glxtr@noidx@displaynumberlist:	
\@glxtr@prevunitcount: new	79	new	91
\@glxtr@setentryunitcountunsetattr:		\@glxtr@noidx@entrynumberlist:	
new	83	new	92
\@glxtr@unitcountlist: new	77	\@glxtr@noidx@numberlistloop:	
\@glxtrpl: new	33	new	92
\@newglossaryentryposthook: added		\@glxtr@reg@glosslist: new	86
empty see value if not set and added		\makeglossaries: new	86
‘see’ to field key map	28	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	31	\glxtrdiscardperiod: added check	
\cGlsformat: added	76	for first use	135
\cGlsformat: added	76	short-desc: fixed typo in	
\cGlsplformat: added	77	\glxtrinlinefullformat and	
\cGlsplformat: added	77	added missing second argument ...	167
\glsdisablehyper: added	61	1.02 (2016-04-25)	
\glsdohyperlink: added	60	\@glxtr@current@style: new	35
\glsdonohyperlink: added	61	\Glsfmtfull: new	208
\glsenableentryunitcount: new	79	\glsfmtfull: new	208
\glshasattribute: added check for		\Glsfmtfullpl: new	209
entry’s existence	116	\glsfmtfullpl: new	209
\glusifattribute: added check for		\Glsfmtlong: new	207
entry’s existence	117	\glsfmtlong: new	207
\glspostlinkhook: added existence		\Glsfmtlongpl: new	208
check	134	\glsfmtlongpl: new	208
\Glsxtr: new	33	\Glsxtrheadfull: new	203

\glxtrheadfull: new	202	\@GLSplural@: set abbreviation and regular format	44
\Glsxtrheadfullpl: new	203	\@GLSsymbol@: set regular format	47
\glxtrheadfullpl: new	202	\@GLSsymbolplural@: set regular format	47
\glxtrheadlong: new	201	\@GLStext@: set abbreviation and regular format	43
\Glsxtrheadlongpl: new	201	\@GLSuseri@: set regular format	48
\glxtrheadlongpl: new	200	\@GLSuserii@: set regular format	48
\Glsxtrtitlefull: new	203	\@GLSuseriii@: set regular format	49
\glxtrtitlefull: new	202	\@GLSuseriv@: set regular format	49
\Glsxtrtitlefullpl: new	203	\@GLSuseriv@: set regular format	49
\glxtrtitlefullpl: new	202	\@GLSuservi@: set regular format	49
\Glsxtrtitlelong: new	201	\@Glsdesc@: set abbreviation and regular format	46
\glxtrtitlelong: new	200	\@Glsdescplural@: set abbreviation and regular format	46
\Glsxtrtitlelongpl: new	202	\@Glsfirst@: set abbreviation and regular format	43
\glxtrtitlelongpl: new	201	\@Glsfirstplural@: set abbreviation and regular format	45
\ifglxtrinsetinside: new	158	\@Glsname@: set abbreviation and regular format	45
postfootnote: added redef of \glxtrsetupfulldefs	164	\@Glsplural@: set abbreviation and regular format	44
stylemods: new	14	\@Glsymbol@: set regular format	47
1.03 (2016-04-27)		\@Glsymbolplural@: set regular format	47
\@GLSfirstplural@: bug fix: misspelt cs name	45	\@Glstext@: set abbreviation and regular format	43
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	44	\@GLSuseri@: set regular format	48
\@Glsfirstplural@: bug fix: misspelt cs name	45	\@GLSuserii@: set regular format	48
\@Glsplural@: fixed bug \@Glsplural@ should be redefined not \@Glsplural	44	\@GLSuseriii@: set regular format	48
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	44	\@GLSuseriv@: set regular format	49
\glxtrtitlelongpl: bug fix: changed \glxtrlong to \glxtrlongpl ..	201	\@GLSuseriv@: set regular format	49
\glxtrtitleshortpl: bug fix: changed \glxtrshort to \glxtrshortpl	196	\@GLSuservi@: set regular format	49
1.04 (2015-04-30)		\@gls@preglossaryhook: added check for entry's existence	133
short-em-footnote: renamed from "footnote-em"	183	\@glsdesc@: set abbreviation and regular format	46
1.04 (2016-05-02)		\@glsdescplural@: set abbreviation and regular format	46
\@glxtrpostloctag: new	38	\@glsfirst@: set abbreviation and regular format	43
\@GLSdesc@: set abbreviation and regular format	46	\@glsfirstplural@: set abbreviation and regular format	44
\@GLSdescplural@: set abbreviation and regular format	46	\@glsname@: set abbreviation and regular format	45
\@GLSfirst@: set abbreviation format ..	44	\@glsplural@: set abbreviation and regular format	44
\@GLSfirstplural@: set abbreviation and regular format	45	\@glsymbol@: set regular format	47
\@GLSname@: set abbreviation and regular format	45		

\@glssymbolplural@: set regular format	47	short-em-nolong-desc: new	182
\@glstext@: set abbreviation and regular format	42	short-em-postfootnote: renamed from “postfootnote-em”	184
\@glstr@deprecated@abbrstyle: new	157	short-footnote: new	164
\@glstr@do@style: new	15	short-long-user: new	191
\@glstr@doloctag: new	38	short-long-user-desc: new	192
\@glstr@idx@entrynumberlist: switched from \let to \newcommand	92	short-nolong: new	167
\@glstr@pagetag: new	38	short-nolong-desc: new	168
\@glstr@pagetag: new	38	short-postfootnote: new	165
\@glstr@preloctag: new	38	short-sc-footnote: renamed from “footnote-sc”	174
\@glstr@postloctag: new	38	short-sc-nolong: new	172
\@glstr@preloctag: new	37, 38	short-sc-nolong-desc: new	173
\glossentrydesc: added glossdescfont attribute check	121	short-sc-postfootnote: renamed from “postfootnote-sc”	174
\Glossentryname: added glossnamefont attribute check	125	short-sm-footnote: renamed from “footnote-sm”	177
\glossentryname: added glossnamefont attribute check	123	short-sm-nolong: new	176
moved post name hook inside condition	125	short-sm-nolong-desc: new	176
\glsabbrvmfont: new	178	short-sm-postfootnote: renamed from “postfootnote-sm”	177
\glsabbrvuserfont: new	185	\letabbreviationstyle: new	157
\glsfirstabbrvmfont: new	178	\newabbreviationstyle: bug fix: corrected test for existence	156
\glsfirstabbrvuserfont: new	185	long-em-noshort-em: new	182
\glsfirstlongemfont: new	178	long-em-noshort-em-desc: new	183
\glsfirstlonguserfont: new	185	long-em-short-em: new	179
\glsifnotregularcategory: new	118	long-em-short-em-desc: new	179
\glslongdefaultfont: new	143	long-noshort: new	170
\glslongemfont: new	178	long-noshort-desc: new	170
\glslongfont: new	143	long-noshort-em: renamed from “long-em”	182
\glslonguserfont: new	185	long-noshort-em-desc: renamed from “long-desc-em”	183
\glstrassignfieldfont: new	42	long-noshort-sc: renamed from “long-sc”	173
\GlsXtrEnablePreLocationTag: new	37	long-noshort-sc-desc: renamed from “long-desc-sc”	173
\glstrfirstscfont: new	171	long-noshort-sm: renamed from “long-sm”	176
\glstrfirstsmfont: new	174	long-noshort-sm-desc: renamed from \long-desc-sm	177
\glstrlongshortdescsort: new	159	long-short-user: new	185
\glstrpostnamehook: added category check	126	long-short-user-desc: new	190
\glstrregularfont: new	39	\renewabbreviationstyle: new style: new	15
\glstruserfield: new	184		
\glstruserparen: new	184	1.05 (2016-06-10)	
\glstrusersuffix: new	185	\eglssetwidest: new	218
\GlsXtrWarnDeprecatedAbbrStyle: new	157	\glsFindWidestAnyName: new	220
short-em-long-em: new	180		
short-em-long-em-desc: new	181		
short-em-nolong: new	181		

\glsFindWidestAnyNameLocation: new	225	\@GLSfirst@: added check for nohyperfirst attribute	44
\glsFindWidestAnyNameSymbol: new	223	\@GLSfirstplural@: added check for nohyperfirst attribute	45
\glsFindWidestAnyNameSymbolLocation: new	224	\@GLSxtrp: new	66
\glsFindWidestLevelTwo: new	221	\@Glsfirst@: added check for nohyperfirst attribute	43
\glsFindWidestUsedAnyName: new	219	\@Glsfirstplural@: added check for nohyperfirst attribute	45
\glsFindWidestUsedAnyNameLocation: new	225	\@Glsxtrp: new	65
\glsFindWidestUsedAnyNameSymbol: new	222	\@gls@preglossaryhook: added \glossxtrsetpopts	134
\glsFindWidestUsedAnyNameSymbolLocation: new	223	\@glsfirst@: added check for nohyperfirst attribute	43
\glsFindWidestUsedLevelTwo: new	220	\@glsfirstplural@: added check for nohyperfirst attribute	44
\glsFindWidestUsedTopLevelName: new	219	\@glsxtrinmark: new	193
\glsfirstlongfootnotefont: new	162	\@glsxtrnotinmark: new	193
\glsgetwidestname: new	218	\@glsxtrp: new	65
\glsgetwidestsubname: new	219	\@glsxtrp@opt: new	65
\glslongfootnotefont: new	162	\glossxtrsetpopts: new	65
\glsxtrAltTreeIndent: new	218	\glsps: new	67
\glsxtralttreeInit: new	218	\glspt: new	67
\glsxtrAltTreePar: new	218	\glsxtr@entry@p: new	66
\glsxtrAltTreeSetHangIndent: new	226	\glsxtrabbrvfootnote: new	162
\glsxtrAltTreeSetSubHangIndent: new	226	\glsxtrchecknohyperfirst: new	43
\glsxtralttreeSubSymbolDescLocation: new	218	\glsxtrfieldtitlecasecs: new	121
\glsxtralttreeSymbolDescLocation: new	217	\glsxtrifinmark: new	193
\glsxtrComputeTreeIndent: new	226	\GLSxtrp: new	69
\glsxtrComputeTreeSubIndent: new	226	\Glsxtrp: new	68
\glsxtrtreeopindent: new	218	\glsxtrp: new	66
short-em-long: fixed incorrect font used by long form	180	\glsxtrsetpopts: new	65
\xglsssetwidest: new	218	short-long-desc: added text key	161
1.06 (2016-06-18)		fixed misspelling of \glsabbrvfont in plural key	161
\@glsdoifexistssorwarn: new	10	long-short-desc: added missing text key	159
\@glsxtr@docdefval: new	9	fixed misspelling of \glsabbrvfont	160
\@glsxtr@usesee: new	29	footnote: changed first forms to use \glsfirstlongfootnotefont	162
General: disabled docdef key at the start of the document	16	postfootnote: removed \footnote from first keys	164
docdef option changed to choice	9	switched from \glsfirstlongfont to \glsfirstlongfootnotefont	165
\glsxtr@usesee: new	29	\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse	85
\glsxtrusesee: new	29	1.08 (2016-12-13)	
\glsxtruseseeformat: new	29	\@@glsxtr@record: new	6
\if@glsxtrdocdefrestricted: new	10		
1.07 (2016-08-15)			
\@@glsxtrp: new	65		

\@GLS@: added \@glxtr@record	41	\@glxtr@redef@forglsentries: new	5
\@GLSpl@: added \@glxtr@record	41	\@glxtr@shortcutsval: new	13
\@Gls@: added \@glxtr@record	40	\@glxtr@unsrt@getgrouptitle: new	103
\@GLspl@: added \@glxtr@record	40	\@print@noidx@glossary: added	
\@Gls@: added \@glxtr@record	40	redefinition	92
\@Gls@@link@: added		\@glxtr@addloclistfield: added	
\@glxtr@record	41	group key	8
\@Gls@field@link: added		added location key	8
\@glxtr@record	39	\@glxtr@fields: new	100
\@Gls@saveentrycounter: new	16	\@glxtr@linkprefix: new	100
\@Glsdisp: added \@glxtr@record	41	\@glxtr@org@newignoredglossary:	
\@GLspl@: added \@glxtr@record	40	new	24
\@glxtr@dorecord: new	7	\@glxtr@s@newignoredglossary: new	24
\@glxtr@err@undefaction: new	5	\@glxtr@shortcutsval: new	100
\@glxtr@record: new	6	\@glxtr@texencoding: new	100
\@glxtr@warn@onexistsordo: new	5	\@glxtr@writefields: new	100
\@glxtr@warn@undefaction: new	5	\@GlsXtrLoadResources: new	99
\@print@unsrt@glossary: new	102	\@glxtr@resourcefile: changed	
General: added record package option	8	extension to .glstex	99
\@glxtr@add: added \@glxtr@record	42	\@newignoredglossary: added starred	
\@glxtr@doifexists: now defines		version	24
\@glxtr@label	27	1.12 (2017-02-03)	
\@glxtr@do@wrglossary: new	16	\@@glxtr@recordcounter: new	7
\@glxtr@addloclistfield: new	8	\@Gls@preglossaryhook: check for	
\@glxtr@indexonly@saveentrycounter:		definition	134
new	7	\@glxtr@counterrecordhook: new	101
\@glxtr@record: new	101	\@glxtr@display@loc: new	93
\@glxtr@resource: new	100	\@glxtr@docounterrecord: new	101
\@glxtr@saveentrycounter: new	16	\@glxtr@longnewglossaryentry:	
\@glxtr@setup@record: new	7	new	23
\@glxtr@assignfieldfont: added check		\@glxtr@noop@recordcounter: new	7
for existence	42	\@glxtr@op@recordcounter: new	7
\@glxtr@resourcefile: new	99	\@glxtr@provide@storagekey: new	17
\@printunsrt@glossaries: new	102	\@glxtr@s@longnewglossaryentry:	
\@printunsrt@glossary: new	101	new	23
1.09 (2016-12-16)		\@glxtr@entryfmt: new	19
\@glxtr@gettype: new	90	\@glxtr@indexaliased: new	57
\@glxtr@mixed@assign@sortkey:		\@glxtr@setaliasnoindex: new	57
new	91	\@newglossaryentryposthook: added	
\@print@glossary: redefined to save		check for alias key	22
options	90	\@no@glxtr@indexaliased: new	58
\@glxtr@makeglossaries: new	90	\@printunsrt@glossary: new	101
1.10 (2016-12-17)		General: added target key to printgloss	
\@GLSpl@: fixed bug caused by typo in		family	90
command name	41	\@aptoglossarypreamble: new	22
1.11 (2017-01-19)		\@csGlsXtrLetField: new	21
\@glxtr@do@redef@forglsentries:		\@eGlsXtrSetField: new	21
new	5	\@gGlsXtrSetField: new	21
\@glxtr@noidx@do: new	103		

\glsdohyperlink: added check for alias field	60	\GlsXtrLetFieldToField: new	21
\glsnoidxdisplayloc: added redefinition	93	\GlsXtrLoadResources: removed restriction on only one per document	99
\glssettoctitle: added patch	25	\glsxtrlocrangefmt: new	94
\glsxtr@counterrecord: new	101	\glsxtrpostlongdescription: new ..	24
\glsxtr@langtag: new	100	\glsxtrprovidestoragekey: new	17
\glsxtr@newabbreviation: new	139	\GlsXtrRecordCounter: new	101
\glsxtr@org@newignoredglossary: Added check for existence	24	\glsxtrresourcecount: new	99
\glsxtr@pluralsuffixes: new	100	\glsxtrresourcefile: added catcode change for @	99
\glsxtr@provideignoredglossary: new	25	\glsxtrsetaliasnoindex: new	57
\glsxtr@s@newignoredglossary: Added check for existence	24	\GlsXtrSetField: new	20
\glsxtr@s@provideignoredglossary: new	26	\glsxtrsetfieldifexists: new	20
\glsxtrabbrvpluralsuffix: new ...	143	\glsxtrunsrtdo: new	103
\glsxtralias: new	22	\GlsXtrusefield: new	20
\glsxtrcopytoglossary: new	27	\glsxtrusefield: new	20
\glsxtrdeffield: new	20	short-postlong-user: new	188
\glsxtrdisplayendloc: new	94	short-postlong-user-desc: new ...	190
\glsxtrdisplayendlochook: new	94	\longnewglossaryentry: added starred version	23
\glsxtrdisplaysingleloc: new	94	long-postshort-user: new	186
\glsxtrdisplaystartloc: new	94	long-postshort-user-desc: new ...	188
\glsxtredeffield: new	20	postdot: new	11
\glsxtrentryfmt: new	18	\pretoglossarypreamble: new	22
\glsxtrfieldddolistloop: new	19	\print@noop@unsrtglossaryunit: new	103
\glsxtrfielddforlistloop: new	20	\print@op@unsrtglossaryunit: new	102
\glsxtrfielddifylist: new	20	\printunsrtglossary: added starred form	101
\glsxtrfieldlistadd: new	19	\printunsrtglossaryhandler: new .	102
\glsxtrfieldlistead: new	19	\printunsrtglossaryunit: new	7
\glsxtrfieldlistgadd: new	19	\printunsrtglossaryunitsetup: new	102
\glsxtrfieldlistxadd: new	19	\provideignoredglossary: new	25
\glsxtrfieldxifylist: new	20	\s@glsxtr@provide@storagekey: new	18
\glsxtrfmt: new	18	\s@printunsrtglossary: new	101
\GlsXtrFmtDefaultOptions: new	18	\xGlsXtrSetField: new	21
\GlsXtrFmtField: new	18	1.13 (2017-02-07)	
\glsxtrifkeydefined: new	17	\@glsdisp: removed	
\glsxtrindexaliased: new	58	\@glsxtr@org@glsdisp	41
\GlsXtrLetField: new	21	\glsxtrsetaliasnoindex: switched to \providecommand	57

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	
\.	135, 217
\@	99
\@@cGLS@	72, 81
\@@cGLSpl@	72, 81
\@@cGlspl@	72, 81
\@@cgls@	72, 81
\@@cglspl@	72, 74, 81
\@@do@wrglossary	7, 87
\@@do@wrglossary	9, 16, 42, 58
\@@glo@assign@sortkey	91
\@@glo@list	5
\@@glo@type	102
\@@gls@expand@field	18
\@@glslocalreset	70
\@@glslocalunset	70
\@@glsreset	70
\@@glsunset	70
\@@glsxtr@autoindex@escspch	129–131
\@@glsxtr@checkspch	128, 129, 131
\@@glsxtr@disabledflycommand	34
\@@glsxtr@record	9
\@@glsxtr@recordcounter	9, 101
\@@glsxtrp	65, 66
\@@glsxtrpostloctag	37
\@@glsxtrpreloctag	37
\@@newglossaryentry@defcounters	71
\@@newglossaryentry@defunitcounters	79
\@@par	218
\@ACRlong	62
\@ACRlongpl	62
\@ACRshort	62
\@ACRshortpl	62
\@Acrlong	62
\@Acrlongpl	62
\@Acrshort	62
\@Acrshortpl	62
\@GLS@	61, 75, 76
\@GLSdesc@	46
\@GLSpl@	61, 75, 76
\@GLSplural@	62
\@GLSsymbol@	48
\@GLStext@	62
\@GLSxtr@full	144
\@GLSxtr@fullpl	146
\@GLSxtr@p@acrlong@	62
\@GLSxtr@p@acrlongpl@	62
\@GLSxtr@p@acrshort@	62
\@GLSxtr@p@acrshortpl@	62
\@GLSxtr@p@long@	62
\@GLSxtr@p@longpl@	62
\@GLSxtr@p@plural@	61
\@GLSxtr@p@short@	61
\@GLSxtr@p@shortpl@	62
\@GLSxtr@p@text@	61
\@GLSxtrlong	62, 149
\@GLSxtrlongpl	62, 152
\@GLSxtrp	69, 70
\@GLSxtrshort	61, 147
\@GLSxtrshortpl	62, 150
\@Gls@	61, 74, 75
\@Gls@acentryname	83
\@Gls@entry@field	54, 68
\@Gls@entryname	83
\@GlsXtrEnableOnTheFly	31, 32
\@Glspl@	61, 75
\@Glsplural@	62
\@Glstext@	62
\@Glsxtr	33, 34
\@Glsxtr@full	144
\@Glsxtr@fullpl	145
\@Glsxtr@p@acrlong@	62
\@Glsxtr@p@acrlongpl@	62
\@Glsxtr@p@acrshort@	62

\@Glsxtr@p@acrshortpl@	62	\@glo@descplural	23
\@Glsxtr@p@long@	61	\@glo@group	8
\@Glsxtr@p@longpl@	62	\@glo@label	8, 17, 22, 28, 29, 54, 219–226
\@Glsxtr@p@plural@	61	\@glo@location	8
\@Glsxtr@p@short@	61	\@glo@loclist	8
\@Glsxtr@p@shortpl@	62	\@glo@name	128
\@Glsxtr@p@text@	61	\@glo@no@assign@sortkey	91
\@Glsxtrlong	61, 148	\@glo@parent	220–222
\@Glsxtrlongpl	62, 152	\@glo@see	22, 28–30
\@Glsxtrp	68	\@glo@sort	128
\@Glsxtrpl	34	\@glo@sorttype	89, 93
\@Glsxtrshort	61, 147	\@glo@text	41
\@Glsxtrshortpl	62, 150	\@glo@thislettergrp	104
\@acrlong	62	\@glo@thisvalue	184
\@acrlongpl	62	\@glo@tmp	17, 54
\@acrshort	62	\@glo@type	29, 83, 86, 89–95, 98, 99, 102
\@acrshortpl	62	\@glo@types	118, 119, 219–225
\@alt@gls@hyp@opt	59	\@glossary@default@style	35, 89, 228
\@auxout	7, 38, 73, 81, 82, 86, 87, 95, 99–101	\@glossarystyle	89, 90
\@bibgls@restoreat	99	\@gls@	61, 74, 75
\@cGLS	76	\@gls@@link	41
\@cGLS@	72, 76, 81	\@gls@actualchar	128
\@cGLSpl	76	\@gls@adjustmode	42
\@cGLSpl@	72, 76, 81	\@gls@alt@hyp@opt	60
\@cGlspl@	72, 81	\@gls@alt@hyp@opt@char	59, 60
\@cgls@	72, 81	\@gls@alt@hyp@opt@keys	59, 60
\@cglspl@	72, 81	\@gls@automake	89
\@disable@onlypremakeg	86	\@gls@checkedmkidx	128, 129, 131
\@do@auxoutstuff	95	\@gls@checkmkidxchars	128
\@do@glssee	22	\@gls@codepage	95
\@do@newglossaryentry	83, 84, 141	\@gls@counter	6, 7, 42, 57
\@do@seeglossary	87	\@gls@currentlettergroup	93, 102, 104
\@empty	42, 50–54, 128, 129, 143–152	\@gls@declareoption	4
\@end@glxtr@addunused	30	\@gls@doautomake	89
\@end@glxtr@gettype	89, 91	\@gls@encapchar	129
\@end@glxtr@usesee	29	\@gls@entry@count	73
\@endfortrue	155	\@gls@entry@field	17, 18, 20, 54, 66–69, 72
\@firstofone	42, 121, 122, 127, 132	\@gls@entry@unitcount	81, 82
\@firstofthree	41, 42, 50–53, 59, 143, 145, 146, 148, 150, 151	\@gls@field@font	42–50
\@firstoftwo	43–48, 51–54, 56, 59, 85, 136, 137, 144–146, 150–152, 193, 194	\@gls@field@link	43–50, 54, 55
\@for	5, 14, 30, 71, 83, 86, 89, 102, 120, 132	\@gls@getgrouptitle	102
\@glo@assign@sortkey	89	\@gls@hyp@opt	55, 60, 76, 143–152
\@glo@category	77	\@gls@hyp@opt@cs	59
\@glo@counterprefix	7	\@gls@increment@currcount	72
\@glo@countunit	77	\@gls@increment@currunitcount	80
\@glo@default@sorttype	89	\@gls@keymap	8, 17, 29, 54, 100
\@glo@desc	23	\@gls@label	7, 59, 87, 101, 155
		\@gls@levelchar	128
		\@gls@link	18, 40, 41, 50–54, 143–152

\@gls@link@checkfirsthyper	41, 85	\@gls@target	61, 90
\@gls@link@nocheckfirsthyper		\@gls@text@	62
.....	40, 50–54, 143–152	\@gls@widestname	218, 219, 226
\@gls@local@increment@currcount	72	\@gls@xtr	32, 34
\@gls@local@increment@currunitcount	80	\@gls@xtr@@do@@wrglossary	87
\@gls@location	103, 104	\@gls@xtr@abbreviationsdef	11, 15, 16
\@gls@loclist	91, 92, 103, 104	\@gls@xtr@activate@initialtagging ..	
\@gls@longpl	138–140	132, 133
\@gls@nohyperlist	24, 26	\@gls@xtr@addunitcounter	77
\@gls@noidx@do	93	\@gls@xtr@addunusedxrefs	29, 30
\@gls@noidx@nosanitizesort	88	\@gls@xtr@attrval	121–128
\@gls@noidx@sanitizesort	88	\@gls@xtr@autoindex@at	128, 129
\@gls@noidx@loclist@finalsep	91	\@gls@xtr@autoindex@doextra@esc	128
\@gls@noidx@loclist@prev	91	\@gls@xtr@autoindex@encap	128, 129
\@gls@noidx@loclist@sep	91	\@gls@xtr@autoindex@esc	128, 130, 131
\@gls@noref@warn	87, 93	\@gls@xtr@autoindex@escat	128, 129
\@gls@org@glsnoidx@displayloc	92	\@gls@xtr@autoindex@escencap	129
\@gls@org@glsseeformat	92	\@gls@xtr@autoindex@esclevel ...	128, 130
\@gls@preglossaryhook	90, 132	\@gls@xtr@autoindex@escquote ...	128, 130
\@gls@prevlevel	227	\@gls@xtr@autoindex@level	128–130
\@gls@quotechar	128	\@gls@xtr@autoindex@setname	127
\@gls@reference	30, 86, 87	\@gls@xtr@autoindex@crossrefs ...	9, 10, 28
\@gls@saveentrycounter	9, 16, 42	\@gls@xtr@cat	71, 83, 132
\@gls@see@noindex	99	\@gls@xtr@counterrecordhook	7
\@gls@setdefault@glslink@opts	58	\@gls@xtr@csname	78–80
\@gls@short	140	\@gls@xtr@current@style	35, 228
\@gls@shortpl	138, 140	\@gls@xtr@currentunitcount	78, 80
\@gls@sort	104	\@gls@xtr@currunitcount	79, 81
\@gls@tmpb	131	\@gls@xtr@declareoption	4, 11, 14
\@gls@type	87, 89, 155, 219–225	\@gls@xtr@defaultnoglossarywarning ..	14
\@gls@write@entrycounts	72	\@gls@xtr@deprecated@abbrstyle	
\@gls@write@entryunitcounts	81	173, 174, 177, 178, 182–184
\@gls@write@entryunitcounts@do	82	\@gls@xtr@disabledflycommand	34
\@gls@abbrv@current@abbreviation	139, 153	\@gls@xtr@display@loc	93
\@gls@acronymlists	83	\@gls@xtr@do@@wrindex	59
\@gls@doifexistsorwarn	10, 123–126	\@gls@xtr@do@glsdisablehyperinlist ..	56
\@gls@entry	73, 81, 82	\@gls@xtr@do@redef@forglsentries	6
\@gls@link	61	\@gls@xtr@do@style	15, 210
\@gls@locref	7	\@gls@xtr@do@titlecaps@warn	
\@gls@nextpages	90	122, 123, 125, 133
\@gls@nonextpages	90	\@gls@xtr@doabbreviationsdef	11
\@gls@numberformat	6, 7, 42, 57, 126, 127	\@gls@xtr@doaccsupp	14, 15
\@gls@order	86	\@gls@xtr@docdefval	10, 31
\@gls@pl@	61, 74, 75	\@gls@xtr@docounterrecord	7
\@gls@plural@	62	\@gls@xtr@doloctag	37, 38
\@gls@punc@token	137	\@gls@xtr@dorecord	7
\@gls@style@alttree	217	\@gls@xtr@dostylewarn	155
\@gls@style@inline	217	\@gls@xtr@enabletagging	132
\@gls@style@listdotted	212	\@gls@xtr@end@	32

\@glxtr@endescspch	128-131	\@glxtr@org@Glsxtrtitlefirstplural	194, 195
\@glxtr@entrycount@org@localreset .	72	\@glxtr@org@Glsxtrtitlefull ..	194, 195
\@glxtr@entrycount@org@localunset .	72	\@glxtr@org@Glsxtrtitlefullpl	194, 195
\@glxtr@entrycount@org@reset	72	\@glxtr@org@Glsxtrtitlelong ..	194, 195
\@glxtr@entrycount@org@unset	72	\@glxtr@org@Glsxtrtitlelongpl	194, 195
\@glxtr@entryunitcount@org@localreset	80	\@glxtr@org@Glsxtrtitleplural	194, 195
\@glxtr@entryunitcount@org@localunset	80	\@glxtr@org@Glsxtrtitleshort .	194, 195
\@glxtr@entryunitcount@org@reset ..	80	\@glxtr@org@Glsxtrtitleshortpl	194, 195
\@glxtr@entryunitcount@org@unset ..	80	\@glxtr@org@Glsxtrtitletext ..	194, 195
\@glxtr@err@undefaction	6, 9	\@glxtr@org@MakeUppercase	194, 195
\@glxtr@field@linkdefs	40	\@glxtr@org@checkfirsthyper	56, 85
\@glxtr@format@overridefalse	127	\@glxtr@org@delimN	37, 38
\@glxtr@format@overridefalse	127	\@glxtr@org@delimR	37, 38
\@glxtr@foundinlist	137	\@glxtr@org@doseeglossary	87
\@glxtr@full	143	\@glxtr@org@gls@	40
\@glxtr@fullpl	145	\@glxtr@org@glsignore	37, 38
\@glxtr@gettype	89	\@glxtr@org@glspl@	40
\@glxtr@glossdescfont	121, 122	\@glxtr@org@glsxtrtitlefirst .	194, 195
\@glxtr@glossnamefont	123-126	\@glxtr@org@glsxtrtitlefirstplural	194, 195
\@glxtr@gobbleto@endescspch	131	\@glxtr@org@glsxtrtitlefull ..	194, 195
\@glxtr@idx@displaynumberlist	88	\@glxtr@org@glsxtrtitlefullpl	194, 195
\@glxtr@idx@entrynumberlist	88	\@glxtr@org@glsxtrtitlelong ..	194, 195
\@glxtr@ifcsstart	32	\@glxtr@org@glsxtrtitlelongpl	194, 195
\@glxtr@ifpunctoken	137	\@glxtr@org@glsxtrtitleplural	194, 195
\@glxtr@ifunitcounter	77	\@glxtr@org@glsxtrtitleshort .	194, 195
\@glxtr@insert@dots	139	\@glxtr@org@glsxtrtitleshortpl	194, 195
\@glxtr@insert@dots@next	139	\@glxtr@org@glsxtrtitletext ..	194, 195
\@glxtr@insert@dots	140	\@glxtr@org@makeglossaries	86
\@glxtr@label	30, 120	\@glxtr@org@markboth	193
\@glxtr@loadstyles	211	\@glxtr@org@markright	193
\@glxtr@longnewglossaryentry	23	\@glxtr@org@newacronymstyle	84, 85
\@glxtr@mixed@assign@sortkey	89	\@glxtr@org@postdescription	134
\@glxtr@noidx@displaynumberlist ...	88	\@glxtr@org@see@noindex	99
\@glxtr@noidx@do	103	\@glxtr@org@setacronymstyle	84, 85
\@glxtr@noidx@entrynumberlist	88	\@glxtr@org@printrglossary	34, 90
\@glxtr@noidx@getgrouptitle	102	\@glxtr@org@warndep	138
\@glxtr@noidx@numberlistloop	88	\@glxtr@p@acrlong@	62
\@glxtr@noop@recordcounter	7, 9	\@glxtr@p@acrlongpl@	62
\@glxtr@notfoundinlist	137	\@glxtr@p@acrshort@	62
\@glxtr@op@recordcounter	9	\@glxtr@p@acrshortpl@	62
\@glxtr@optlist	34	\@glxtr@p@long@	61
\@glxtr@org@GLS@	41	\@glxtr@p@longpl@	62
\@glxtr@org@GLSpl@	41	\@glxtr@p@plural@	61
\@glxtr@org@Gls@	40	\@glxtr@p@short@	61
\@glxtr@org@Glspl@	40	\@glxtr@p@shortpl@	62
\@glxtr@org@Glsxtrtitlefirst .	194, 195	\@glxtr@p@text@	61
		\@glxtr@pagetag	37, 38

<code>\@glxtr@pagetag</code>	37, 38	<code>\@ifstar</code>	17, 23–25, 31, 59, 101, 132
<code>\@glxtr@prevunitcount</code>	79	<code>\@ifundefined</code>	209
<code>\@glxtr@printglossopts</code>	34, 89, 90	<code>\@ignored@glossaries</code>	24–26
<code>\@glxtr@provide@addstoragekey</code>	18	<code>\@input</code>	99
<code>\@glxtr@provide@storagekey</code>	17	<code>\@input@</code>	94
<code>\@glxtr@record</code>	9, 40–42	<code>\@istfilename</code>	86
<code>\@glxtr@redef@forglentries</code>	6, 15, 16	<code>\@makeglossary</code>	86
<code>\@glxtr@redefstyles</code>	14, 15, 210	<code>\@mfu@domakefirstuc</code>	132, 133
<code>\@glxtr@reg@glosslist</code>	86–89, 91	<code>\@mfu@nocaplist</code>	133
<code>\@glxtr@s@longnewglossaryentry</code>	23	<code>\@one</code>	73, 82
<code>\@glxtr@savepreloctag</code>	37, 38	<code>\@newglossaryentry@defcounters</code>	71, 79
<code>\@glxtr@setentrycountunsetattr</code>	71	<code>\@newglossaryentryposthook</code>	8, 17, 54
<code>\@glxtr@setentryunitcountunsetattr</code>	82	<code>\@newglossaryentryprehook</code>	8, 17, 23, 54
<code>\@glxtr@setupshortcuts</code>	13, 15, 16	<code>\@nil</code>	104
<code>\@glxtr@shortcutsval</code>	13, 100	<code>\@nnil</code>	128, 129, 131, 137, 139
<code>\@glxtr@swaptwo</code>	138	<code>\@no@glxtrindexaliased</code>	57, 58
<code>\@glxtr@tag</code>	132	<code>\@no@makeglossaries</code>	98
<code>\@glxtr@taggingcs</code>	132	<code>\@nopostdesc</code>	90
<code>\@glxtr@thisloctag</code>	38	<code>\@onelevel@sanitize</code>	34
<code>\@glxtr@tmp</code>	14, 15	<code>\@onlypreamble</code>	35, 37, 81, 99, 101, 127, 129, 130, 132
<code>\@glxtr@type</code>	120	<code>\@org@glossaryentrynumbers</code>	89, 90
<code>\@glxtr@unitcountlist</code>	77	<code>\@org@newglossaryentryprehook</code>	23
<code>\@glxtr@unsrt@getgrouptitle</code>	102	<code>\@print@unsrt@glossary</code>	101
<code>\@glxtr@usesee</code>	29	<code>\@printgloss@setsort</code>	89, 90
<code>\@glxtr@warn@onexistsordo</code>	6, 9	<code>\@printglossary</code>	34, 101
<code>\@glxtr@warn@undefaction</code>	6, 9	<code>\@printunsrtglossary</code>	101
<code>\@glxtrdocdeffalse</code>	31	<code>\@sGlsXtrEnableOnTheFly</code>	31
<code>\@glxtrentryfmt</code>	19	<code>\@secondofthree</code>	43–45, 50–53, 55, 144, 145, 147, 148, 150, 152
<code>\@glxtrindexaliased</code>	57	<code>\@secondoftwo</code>	41, 42, 45–54, 56, 61, 85, 90, 137, 143, 144, 146–152, 164, 193, 195
<code>\@glxtrindexcrossrefsfalse</code>	10	<code>\@sglxtr@provide@storagekey</code>	17
<code>\@glxtrindexcrossrefstrue</code>	10	<code>\@thirdofthree</code>	43–45, 51–55, 144, 146, 147, 149, 151, 152
<code>\@glxtrinmark</code>	193	<code>\@thirdoftwo</code>	46–50
<code>\@glxtrlong</code>	61, 148	<code>\@warn@nomakeglossaries</code>	95
<code>\@glxtrlongpl</code>	62, 151	<code>\@xdy@main@language</code>	95
<code>\@glxtrnotinmark</code>	193	<code>\@xdylanguage</code>	95
<code>\@glxtrp</code>	67	<code>_</code>	97
<code>\@glxtrp@opt</code>	65		
<code>\@glxtrpl</code>	33, 34		
<code>\@glxtrpostloctag</code>	36, 37, 39		
<code>\@glxtrpreloctag</code>	36, 37, 39		
<code>\@glxtrsetaliasnoindex</code>	57, 58		
<code>\@glxtrshort</code>	61, 146		
<code>\@glxtrshortpl</code>	62, 149		
<code>\@glxtrundeftag</code>	5, 16		
<code>\@gobble</code>	6, 9, 11, 42, 139		
<code>\@gobbletwo</code>	138		
<code>\@ifnextchar</code>	59		
<code>\@ifpackageloaded</code>	4, 11, 100, 105, 121, 123, 125, 127, 210		

A

<code>\AB</code>	12
<code>\Ab</code>	12
<code>\ab</code>	11
abbreviation styles:	
long-noshort	182, 183
long-postshort-user	188

long-short-user	186		
short	167		
short-long-user	188		
short-postlong-user	190		
\abbreviationsname	11		
\abbrvpluralsuffix			
.....	100, 140, 158, 160, 163,		
	165–167, 169, 171–178, 186, 187, 189, 191		
\ABP	12		
\Abp	12		
\abp	12		
\ACRfullfmt	84		
\Acrfullfmt	84		
\acrfullfmt	84		
\ACRfullplfmt	84		
\Acrfullplfmt	84		
\acrfullplfmt	84		
\acronymentry	83		
\acronymfont	50–54, 64, 84, 85		
\acronymname	11		
\acronymsort	83		
\acronymtype	11, 83, 84		
\acrpluralsuffix	84, 100		
\actualchar	130		
\addtolength	227		
\advance	73, 82, 99		
\AF	12		
\Af	12		
\af	12		
\AFP	12		
\Afp	12		
\afp	12		
\AL	12		
\Al	12		
\al	12		
\ALP	12		
\Alp	12		
\alp	12		
\AnyTrackedLanguages	210		
\appto	8, 15, 17,		
	22, 28, 29, 54, 59, 71, 79, 103, 127, 136, 139		
\AS	12		
\As	12		
\as	12		
\ASP	12		
\Asp	12		
\asp	12		
\AtBeginDocument	16, 35, 36, 101		
\AtEndDocument	29, 72, 81, 95		
		B	
		babel package	127, 129, 136
		\begin	93, 96, 97, 102, 212–216
		\begingroup	6, 57, 101
		\bgroup	23, 89
		C	
		\catcode	99
		category attributes:	
		discardperiod	135
		entrycount	70, 71, 73, 82
		firstuc	125
		glossdesc	121
		glossdescfont	121
		glossname	123
		glossnamefont	123, 125
		headuc	195
		indexname	128
		indexonlyfirst	58
		insertdots	140
		nohyper	56
		nohyperfirst	43–45
		regular	39, 76, 158,
			160–162, 164, 167–169, 179, 181, 185, 191
		\cGLS	12, 71, 82
		\cGLs	12, 71, 82
		\cglS	11, 71, 82
		\cGLSformat	75
		\cGlsformat	74
		\cglSformat	74, 76
		\cGLSpl	12, 71, 82
		\cGlspl	12, 71, 82
		\cglSpl	12, 71, 82
		\cGLSplformat	75
		\cGlsplformat	75
		\cglSplformat	74, 76
		\columnwidth	35, 36
		\count@	73, 82
		\csappto	22
		\csdef	17, 20, 22, 54, 55, 72, 77, 78,
			80, 115, 155–157, 164, 186, 188–190, 212
		\cseappto	27
		\csedef	20, 79
		\csgdef	21, 24–26, 30, 31, 37, 72, 73, 78, 80, 81
		\cslet	21, 23, 90
		\csletcs	21, 157
		\csname	5, 6, 18, 25, 35, 39, 41, 42,
			50–55, 57, 65, 78, 79, 87, 94, 95, 98, 99,
			102, 104, 121, 138, 143–153, 157, 158, 226
		\cspreto	22

<code>\csuse</code>	18, 19, 22, 25, 37, 54, 55, 67, 68, 77–81, 89, 93, 94, 101, 103, 116, 127, 134, 135, 155–157, 219–222	127–129, 131, 137, 139, 146–152, 159, 161, 163–170, 186–192, 213–217, 227, 228
<code>\csxdef</code>	28, 78, 81	
<code>\currentglossary</code>	90	
<code>\CurrentOption</code>	15, 211	
<code>\CurrentTrackedLanguageTag</code>	100	
<code>\CurrentTrackedTag</code>	210	
<code>\CustomAbbreviationFields</code>	141, 158–162, 164, 166, 167, 169, 170, 179, 180, 182, 185, 186, 188, 190, 191	
D		
<code>\DeclareAcronymList</code>	83	
<code>\DeclareOption</code>	4, 211	
<code>\DeclareOptionX</code>	4, 15	
<code>\def</code>	6, 8, 9, 16, 23, 25, 29, 30, 32–34, 36, 39–54, 57, 60–65, 74–76, 83, 87, 89–91, 93, 94, 102, 103, 126, 128–133, 137–140, 143–152, 155, 227	
<code>\defglentryfmt</code>	24–26	
<code>\define@boolkey</code>	10, 57	
<code>\define@choicekey</code>	6, 8, 10, 13, 14, 90	
<code>\define@key</code>	8, 14, 15, 17, 54, 138	
<code>\DefineAcronymSynonyms</code>	13	
<code>\delimN</code>	37, 38	
<code>\delimR</code>	37, 38	
<code>\detokenize</code>	32	
<code>\dimen@</code>	85, 219–226	
<code>\dimen@i</code>	220–222	
<code>\dimen@ii</code>	220–222	
<code>\dimexpr</code>	35, 36, 218	
<code>\disable@keys</code>	11, 16, 31	
<code>\do</code>	5, 14, 30, 71, 83, 86, 89, 102, 120, 132	
<code>\do@gl@link@checkfirsthyper</code>	18, 40, 41, 50–54, 143–152	
<code>\do@gl@disablehyperinlist</code>	57	
<code>doc</code> package	130	
<code>\dolistcsloop</code>	19	
<code>\DTLifinlist</code>	87, 88, 91	
E		
<code>\eappto</code>	7, 14, 24–26, 128, 211	
<code>\edef</code>	5, 6, 22, 24–27, 42, 56, 57, 77, 78, 80, 86, 87, 91, 94, 95, 99, 121–126, 128, 129, 131, 138, 220–222	
<code>\eglssetwidest</code>	219–226	
<code>\egroup</code>	23, 24, 90	
<code>\else</code>	6, 7, 10, 11, 13, 14, 21, 31, 32, 37, 39, 41, 57, 58, 74, 85, 88, 90, 93, 96–99,	
<code>\emph</code>	178	
<code>\empty</code>	93, 94	
<code>\encapchar</code>	130	
<code>\end</code>	93, 96, 97, 102, 212–216	
<code>\end@gl@xtr@display@loc</code>	93	
<code>\endcsname</code>	5, 6, 18, 25, 35, 39, 41, 42, 50–55, 57, 65, 78, 79, 87, 94, 95, 98, 99, 102, 104, 121, 138, 143–153, 157, 158, 226	
<code>\endgroup</code>	7, 58, 101	
entry categories:		
abbreviation	153	
general	115, 117	
index	119	
<code>\epreto</code>	128	
<code>\equal</code>	98	
etoolbox package	4	
<code>\expandafter</code>	15, 18, 29, 30, 32–34, 54, 55, 59, 65, 76, 77, 87–89, 91, 94, 102, 104, 121, 124, 125, 128, 130, 131, 137, 140	
<code>\expandonce</code>	83, 128, 129	
F		
<code>\fi</code> 6, 7, 9–11, 13, 14, 21, 29, 31, 32, 35–37, 39, 41, 58, 73, 74, 81, 82, 85, 88–90, 93–99, 127–129, 131, 137, 139, 146–152, 159, 161, 163–170, 186–192, 213–217, 219–228		
first use	229	
flag	229	
text	229	
<code>\firstacronymfont</code>	84, 85	
fontspec package	100	
<code>\footnote</code>	162	
<code>\forallglossaries</code>	29, 102, 118–120, 219–225	
<code>\forallgl@entries</code>	73, 82	
<code>\ForEachTrackedDialect</code>	210	
<code>\for@gl@entries</code>	5, 29, 118–120, 219–225	
<code>\forlistcsloop</code>	20, 82, 93	
<code>\forlistloop</code>	91, 92, 133	
<code>\futurelet</code>	137	
G		
<code>\gdef</code>	38, 129, 130	
<code>\Genacrfullformat</code>	84	
<code>\genacrfullformat</code>	84	
<code>\GenericAcronymFields</code>	84	
<code>\Genplacrfullformat</code>	84	
<code>\genplacrfullformat</code>	84	
<code>\glo@grabfirst</code>	104	

<code>\glo@name</code>	124, 125	<code>\gls@noidxglossary</code>	87
<code>\gloaliaslabel</code>	60, 61	<code>\gls@org@glossaryentryfield</code>	90
<code>\global</code>	23, 90, 103	<code>\gls@org@glossarysubentryfield</code>	90
<code>\glolinkprefix</code>	61, 101, 103	<code>\gls@save@numberlist</code>	36, 37, 39
<code>glossaries package</code>	211	<code>\gls@tmplen</code>	219–225, 227
<code>glossaries-accsupp package</code>	14, 15, 104	<code>\gls@type</code>	87
<code>glossaries-extra package</code>	2	<code>\glsabbrvdefaultfont</code>	
<code>glossaries-extra-stylemods package</code> .	14, 134, 210	142, 159, 160, 163, 165–167, 169
<code>glossaries.sty package</code>	23	<code>\glsabbrvmfont</code>	178–184
<code>\GlossariesExtraWarning</code>		<code>\glsabbrvfont</code>	62, 63, 84, 142, 146,
.....	5, 11, 22, 32, 34, 85,	147, 150, 151, 153, 154, 158–167, 169–191	
87, 94, 97, 99, 102, 121–126, 132, 133, 157		<code>\glsabbrvuserfont</code>	185–187, 189, 191
<code>\GlossariesExtraWarningNoLine</code> .	11, 73, 82	<code>\glsabbrvfont</code>	158, 160,
<code>\GlossariesWarning</code> ..	37, 87, 89, 91, 92, 155	162, 164, 179, 180, 185, 186, 188, 190, 191	
<code>\GlossariesWarningNoLine</code>	87, 95	<code>\GLSaccessdesc</code>	46
<code>glossary styles:</code>		<code>\Glsaccessdesc</code>	46, 121, 131
<code>alttree</code>	217, 218, 226	<code>\glsaccessdesc</code>	46, 122, 135
<code>inline</code>	217	<code>\GLSaccessdescplural</code>	47
<code>listdotted</code>	212	<code>\Glsaccessdescplural</code>	46
<code>listdottedstyle</code>	212	<code>\glsaccessdescplural</code>	46
<code>sublistdotted</code>	212	<code>\GLSaccessfirst</code>	44
<code>glossary-long package</code>	213	<code>\Glsaccessfirst</code>	43
<code>glossary-longbooktabs package</code>	213	<code>\glsaccessfirst</code>	43
<code>glossary-mcols package</code>	211	<code>\GLSaccessfirstplural</code>	45
<code>glossary-tree package</code>	211	<code>\Glsaccessfirstplural</code>	45
<code>\glossaryentrynumbers</code> .	39, 89, 90, 103, 104	<code>\glsaccessfirstplural</code>	45
<code>\glossaryheader</code>	93, 102, 212–216, 227	<code>\Glsaccesslong</code>	
<code>\glossaryname</code>	89	52, 141, 149, 159, 166, 169, 186, 187
<code>\glossarypostamble</code>	93, 102, 103	<code>\glsaccesslong</code>	
<code>\glossarypreamble</code>	93, 102	52, 53, 141, 148, 149, 159, 161,
<code>\glossarysection</code>	93, 98, 102, 103	163, 165, 166, 168–170, 186, 187, 189–192	
<code>\glossarytitle</code>	25, 89, 90, 93, 98, 102	<code>\Glsaccesslongpl</code>	
<code>\glossarytoctitle</code>	25, 89, 93, 98, 102	53, 142, 152, 159, 166, 169, 186–188
<code>\glossentry</code>	90, 104, 212–216, 227	<code>\glsaccesslongpl</code> 53, 54, 141, 151, 152, 159,	
<code>\glossentrydesc</code>	212–218	161, 163–166, 168–170, 186, 187, 190, 192	
<code>\glossentryname</code>	212–216, 227	<code>\GLSaccessname</code>	46
<code>\glossentrysymbol</code>	213–218	<code>\Glsaccessname</code>	45
<code>\glossxtrsetpopts</code>	134	<code>\glsaccessname</code>	45
<code>\GLS</code>	71, 82	<code>\GLSaccessplural</code>	44
<code>\Gls</code>	33, 71, 82	<code>\Glsaccessplural</code>	44
<code>\gls</code>	21, 33, 34, 71, 82, 87, 96	<code>\glsaccessplural</code>	44
<code>\gls@assign@desc</code>	23	<code>\Glsaccessshort</code>	50,
<code>\gls@assign@field</code>	8, 17, 54	147, 153, 161, 163, 165, 168, 189, 190, 192	
<code>\gls@checkseeallowed</code>	31, 86	<code>\glsaccessshort</code>	
<code>\gls@codepage</code>	95	50, 51, 141, 146, 147, 153, 154,
<code>\gls@defdocnewglossaryentry</code>	71, 79	159, 161, 163, 165–169, 186, 187, 189, 191	
<code>\gls@defglossaryentry</code>	23, 24, 33, 34	<code>\Glsaccessshortpl</code>	51,
<code>\gls@dotocitle</code>	89, 90	150, 153, 161, 163–165, 168, 189, 190, 192	
<code>\gls@level</code>	103, 104		

<code>\glsaccesssshortpl</code>	51, 52, 141, 142, 150, 151, 153, 159, 161, 163, 165–169, 186–191
<code>\GLSaccessssymbol</code>	47
<code>\Glsaccessssymbol</code>	47, 131
<code>\glsaccessssymbol</code>	47, 131, 135
<code>\GLSaccessssymbolplural</code>	48
<code>\Glsaccessssymbolplural</code>	47
<code>\glsaccessssymbolplural</code>	47
<code>\GLSaccesstext</code>	43
<code>\Glsaccesstext</code>	43
<code>\glsaccesstext</code>	43
<code>\glsacrshortcutstrue</code>	13
<code>\glsacspacemax</code>	85
<code>\glsadd</code>	30, 96
<code>\glsaddstoragekey</code>	22, 115
<code>\glsbackslash</code>	32
<code>\glscapscase</code>	41–55, 143–154
<code>\glscategory</code>	39, 42, 56, 62, 63, 116–118, 121–127, 131, 134, 135, 143–147, 149–151
<code>\glscategorylabel</code>	56, 138–140, 164, 186, 188–190
<code>\glsclosebrace</code>	97, 98
<code>\glscurrententrylabel</code>	36–38, 90, 102, 127, 133, 134
<code>\glscurrentfieldvalue</code>	18, 19, 184
<code>\glscustomtext</code>	40, 41, 50–54, 143–153, 155
<code>\glsdefaulttype</code>	5, 11, 22, 89, 96, 101, 102
<code>\glsdescriptionaccessdisplay</code>	109, 122
<code>\glsdescriptionpluralaccessdisplay</code>	109, 110
<code>\glsdescwidth</code>	212, 214–216
<code>\glsdetoklabel</code>	19–21, 23, 27–32, 42, 57, 61, 72, 73, 78–82, 87, 90–92, 103, 104, 121, 124, 125, 220–222
<code>\glsdisplaynumberlist</code>	88, 91
<code>\glsdohyperlink</code>	61
<code>\glsdohypertarget</code>	90
<code>\glsdoifexists</code>	10, 20, 27, 29, 40–42, 50–54, 91, 92, 143–152
<code>\glsdoifexistsordo</code>	18, 19, 41
<code>\glsdoifexistsorwarn</code>	10, 121, 122, 131
<code>\glsdoifnoexists</code>	23
<code>\glsdonohyperlink</code>	61
<code>\glsdosanitizesort</code>	88
<code>\glsenableentrycount</code>	71, 73, 81
<code>\glsenableentryunitcount</code>	73, 82
<code>\glsentrycurrcount</code>	72, 73, 79
<code>\Glsentrydesc</code>	109, 114, 122
<code>\glsentrydesc</code>	109, 114, 122
<code>\Glsentrydescplural</code>	110, 114
<code>\glsentrydescplural</code>	109, 110, 114
<code>\Glsentryfirst</code>	76, 107, 113
<code>\glsentryfirst</code>	76, 106, 107, 113, 206
<code>\Glsentryfirstplural</code>	77, 107, 113
<code>\glsentryfirstplural</code>	77, 107, 113, 207
<code>\glsentryfmt</code>	24–26
<code>\Glsentryfull</code>	84
<code>\glsentryfull</code>	84
<code>\Glsentryfullpl</code>	84
<code>\glsentryfullpl</code>	84
<code>\glsentryitem</code>	212–216, 227
<code>\Glsentrylong</code>	63, 64, 76, 111, 115
<code>\glsentrylong</code>	63, 64, 76, 111, 114, 115, 164, 189, 190, 207
<code>\Glsentrylongpl</code>	64, 77, 112, 115
<code>\glsentrylongpl</code>	64, 65, 77, 111, 112, 115, 208
<code>\Glsentryname</code>	105, 112, 123, 124, 126
<code>\glsentryname</code>	105, 112, 128, 219–226
<code>\glsentrynumberlist</code>	88, 92, 224–226
<code>\Glsentryplural</code>	106, 112
<code>\glsentryplural</code>	106, 112, 113, 205, 206
<code>\glsentryprevcount</code>	72, 73, 79
<code>\glsentryprevmaxcount</code>	80
<code>\glsentryprevtotalcount</code>	80
<code>\Glsentryshort</code>	63, 64, 110, 114
<code>\glsentryshort</code>	62–64, 85, 110, 114, 186, 188, 204
<code>\Glsentryshortpl</code>	63, 64, 111, 114
<code>\glsentryshortpl</code>	63, 64, 111, 114, 204, 205
<code>\Glsentrysymbol</code>	108, 113
<code>\glsentrysymbol</code>	108, 113, 223, 224
<code>\Glsentrysymbolplural</code>	108, 113
<code>\glsentrysymbolplural</code>	108, 109, 113
<code>\Glsentrytext</code>	106, 112
<code>\glsentrytext</code>	105, 106, 112, 205
<code>\Glsentryuseri</code>	48
<code>\glsentryuseri</code>	48
<code>\Glsentryuserii</code>	48
<code>\glsentryuserii</code>	48
<code>\Glsentryuseriii</code>	48
<code>\glsentryuseriii</code>	49
<code>\Glsentryuseriv</code>	49
<code>\glsentryuseriv</code>	49
<code>\Glsentryuserv</code>	49
<code>\glsentryuserv</code>	49
<code>\Glsentryuservi</code>	49
<code>\glsentryuservi</code>	50
<code>\glsfieldfetch</code>	60

<code>\glsfieldxdef</code>	120	<code>\glskeylisttok</code>	83, 84, 139, 141
<code>\glsfindwidesttoplevelname</code>	219	<code>\glslabel</code>	27, 39, 56–58, 60, 85, 134, 135, 153, 154, 164, 186, 188–190
<code>\GLSfirst</code>	199	<code>\glslabeltok</code> .	83, 139, 141, 158–162, 164, 166, 167, 169, 170, 179–182, 185, 187–191
<code>\Glsfirst</code>	199	<code>\glsletentryfield</code>	128
<code>\glsfirst</code>	199	<code>\glslink</code>	84
<code>\glsfirstabbrvdefaultfont</code>		<code>\glslink options</code>	
.....	142, 159, 160, 163, 165–167, 169	format	127
<code>\glsfirstabbrvfont</code>		hyper	192
.....	84, 141, 142, 158–169, 171–192	noindex	6, 56, 192
<code>\glsfirstabbrvuserfont</code>	186–191	<code>\glslinkcheckfirsthyperhook</code>	56
<code>\glsfirstaccessdisplay</code>	106, 107	<code>\glslinkvar</code>	59
<code>\glsfirstlongdefaultfont</code>		<code>\glslistdottedwidth</code>	212
.....	159, 160, 166, 168, 169	<code>\glslocalunset</code>	41
<code>\glsfirstlongemfont</code>	179–181, 183	<code>\glslongaccessdisplay</code>	111
<code>\glsfirstlongfont</code> ...	141, 142, 158–161, 163, 165, 166, 168–170, 179–183, 185–192	<code>\glslongdefaultfont</code>	
<code>\glsfirstlongfootnotefont</code>	162–165	143, 159, 160, 162, 166, 168, 169
<code>\glsfirstlonguserfont</code> ..	186, 187, 189, 191	<code>\glslongemfont</code>	178–183
<code>\GLSfirstplural</code>	199, 200	<code>\glslongfont</code>	63, 64, 143, 148, 149, 151, 152, 159, 160, 163, 165, 166, 168, 169, 179–181, 183, 186, 187, 189, 191
<code>\Glsfirstplural</code>	200	<code>\glslongfootnotefont</code>	162, 163, 165
<code>\glsfirstplural</code>	199, 200	<code>\glslongpltok</code> ...	140, 141, 158–162, 169, 170, 179, 180, 182, 185, 186, 188, 190, 191
<code>\glsfirstpluralaccessdisplay</code>	107	<code>\glslongpluralaccessdisplay</code> ...	111, 112
<code>\glsforeachincategory</code>	155	<code>\glslongtok</code>	83, 84, 139, 141, 158–162, 164, 166, 167, 169, 170, 179, 180, 182, 185, 186, 188, 190, 191
<code>\glsgenentryfmt</code>	39	<code>\glslonguserfont</code>	185–189, 191
<code>\glsgetattribute</code>	60, 73, 78, 79, 121–127	<code>\glsnameaccessdisplay</code> ..	105, 123, 124, 126
<code>\glsgetcategoryattribute</code>	116	<code>\glsnamefont</code>	123–126
<code>\glsgetwidestname</code>	218	<code>\glsnextpages</code>	90
<code>\glsgroupheading</code>	104, 212–216, 227	<code>\glsnoidxdisplayloc</code>	92
<code>\glsgroupskip</code>	104, 213–217, 228	<code>\glsnoidxdisplaylocclishandler</code>	91
<code>\glschasattribute</code>		<code>\glsnoidxloclist</code>	92, 103, 104
.....	60, 73, 78, 80, 82, 121–127, 158, 160–162, 164, 179, 181, 185, 187–191	<code>\glsnoidxnumberlistloophandler</code>	92
<code>\glschascategoryattribute</code>	116	<code>\glsnonextpages</code>	90
<code>\glshypernumber</code>	127	<code>\glsnonumberlistfalse</code>	36
<code>\glsifattribute</code>	43, 56, 58, 66, 119, 121–125, 133, 136, 195–203	<code>\glsnonumberlisttrue</code>	37
<code>\glsifcategory</code>	118	<code>\glsnumberlistloop</code>	88
<code>\glsifcategoryattribute</code> .	56, 117, 118, 140	<code>\glsnumlistlastsep</code>	91
<code>\glsifnotregular</code>	42	<code>\glsnumlistsep</code>	91
<code>\glsifnotregularcategory</code>	118	<code>\glsopenbrace</code>	97, 98
<code>\glsifplural</code> 41, 42, 44–48, 50–54, 136, 143–154		<code>\glsorder</code>	86
<code>\glsifregular</code>	39, 42, 76, 77	<code>\glspagelistwidth</code>	212, 214–216
<code>\glsifregularcategory</code>	118	<code>\glspar</code>	103
<code>\glsifusetranslator</code>	25	<code>\GLSpl</code>	71, 82
<code>\glsignore</code>	37, 38	<code>\Glspl</code>	34, 71, 82
<code>\glsinlinedescformat</code>	217		
<code>\glsinlinesubdescformat</code>	217		
<code>\glsinsert</code>	41, 42, 50–54, 143–154		

<code>\glspl</code>	33, 71, 82	<code>\glsxtr@addloclistfield</code>	9
<code>\GLSplural</code>	198	<code>\glsxtr@addunused</code>	30
<code>\Glsplural</code>	198	<code>\glsxtr@applyabbrvfmt</code>	153
<code>\glsplural</code>	198	<code>\glsxtr@applyabbrvstyle</code>	138, 139, 155
<code>\glspluralaccessdisplay</code>	106	<code>\glsxtr@counterrecord</code>	101
<code>\glspluralsuffix</code>	100, 139, 143	<code>\glsxtr@doption</code>	4, 11, 15, 16
<code>\glspostdescription</code>	134, 212–218	<code>\glsxtr@fields</code>	100
<code>\glspostinline</code>	217	<code>\glsxtr@headentry@p</code>	66, 67
<code>\glspostlinkhook</code> .	40, 41, 50–54, 65, 143–152	<code>\glsxtr@ifnextpunc</code>	137
<code>\glsprestandardsort</code>	88	<code>\glsxtr@ifpunctoken</code>	137
<code>\glsresetentrylist</code>	93, 102	<code>\glsxtr@indexonly@saveentrycounter</code>	9, 16
<code>\glssee</code>	22	<code>\glsxtr@keylist</code>	32–34
<code>\glsseeformat</code>	29, 87, 92	<code>\glsxtr@langtag</code>	100
<code>\glssetabbrvfmt</code>	39,	<code>\glsxtr@linkprefix</code>	101
	42, 62, 63, 121–126, 131, 143–147, 149–151	<code>\glsxtr@makeglossaries</code>	86
<code>\glssetattribute</code>	158, 160–162, 164, 166,	<code>\glsxtr@newabbreviation</code>	84, 139
	167, 169, 170, 179, 181, 182, 185, 187–191	<code>\glsxtr@next</code>	137
<code>\glssetcategoryattribute</code>		<code>\glsxtr@org@newignoredglossary</code>	24
	71, 83, 85, 116, 117, 119, 120, 132	<code>\glsxtr@orgmakenoidxglossaries</code>	30
<code>\glssetnoexpandfield</code>	8	<code>\glsxtr@pluralsuffixes</code>	100
<code>\glssettoctitle</code>	89	<code>\glsxtr@provideignoredglossary</code>	25
<code>\glsshortaccessdisplay</code>	110	<code>\glsxtr@punclist</code>	136, 137
<code>\glsshortpltok</code> ..	140, 141, 158–162, 164,	<code>\glsxtr@record</code>	7
	166, 167, 179, 180, 185, 186, 188, 190, 191	<code>\glsxtr@resource</code>	99
<code>\glsshortpluralaccessdisplay</code> ..	110, 111	<code>\glsxtr@s@newignoredglossary</code>	24
<code>\glsshorttok</code>	83,	<code>\glsxtr@s@provideignoredglossary</code> ...	25
	84, 139–141, 158–162, 164, 166, 167,	<code>\glsxtr@saveentrycounter</code>	6, 7, 57
	170, 179, 180, 182, 185, 186, 188, 190, 191	<code>\glsxtr@setup@record</code>	8, 9, 16
<code>\glssubentryitem</code>	212–217, 227	<code>\glsxtr@shortcutsval</code>	100
<code>\glssymbolaccessdisplay</code>	108	<code>\glsxtr@texencoding</code>	100
<code>\glssymbolpluralaccessdisplay</code> .	108, 109	<code>\glsxtr@usesee</code>	29
<code>\glstarget</code>	212–217, 227	<code>\glsxtr@warnonexistsordo</code>	6, 9, 28
<code>\GLStext</code>	197	<code>\glsxtr@writefields</code>	99
<code>\Glstext</code>	197, 198	<code>\glsxtrabbrvfootnote</code>	162–164
<code>\glstext</code>	197	<code>\glsxtrabbrvpluralsuffix</code> ...	100, 143,
<code>\glstextaccessdisplay</code>	105, 106		158, 160, 163, 165–167, 169, 171, 175, 185
<code>\glstextformat</code>	41	<code>\glsxtrabbrvtype</code>	11, 141
<code>\glstextup</code>	171	<code>\glsxtraddallcrossrefs</code>	29
<code>\glstreeindent</code>	226, 227	<code>\glsxtralias</code>	58
<code>\glstreenamebox</code>	227	<code>\glsxtrAltTreeIndent</code>	218
<code>\glstreenamefmt</code>	218–227	<code>\glsxtralttreeInit</code>	226
<code>\GlstrLetField</code>	21	<code>\glsxtrAltTreePar</code>	217
<code>\glstype</code>	41, 50–54, 143–152	<code>\glsxtrAltTreeSetHangIndent</code> ...	218, 227
<code>\glunset</code>	30, 41, 74, 75	<code>\glsxtrAltTreeSetSubHangIndent</code>	227
<code>\glswrite</code>	86	<code>\glsxtralttreeSubSymbolDescLocation</code>	227
<code>\glswriteentry</code>	6	<code>\glsxtralttreeSymbolDescLocation</code> ..	
<code>\Glsxtr</code>	34		218, 227
<code>\glsxtr</code>	34	<code>\glsxtrassignfieldfont</code>	43–50
<code>\glsxtr@do@wrglossary</code>	7, 9	<code>\glsxtrcat</code>	33, 34

<code>\glxtrchecknohyperfirst</code>	43–45	<code>\Glsxtrheadfullpl</code>	194
<code>\glxtrComputeTreeIndent</code>	227	<code>\glxtrheadfullpl</code>	194
<code>\glxtrComputeTreeSubIndent</code>	227	<code>\Glsxtrheadlong</code>	194
<code>\GlsXtrDefineAbbreviationShortcuts</code> .	13	<code>\glxtrheadlong</code>	194
<code>\GlsXtrDefineOtherShortcuts</code>	13	<code>\Glsxtrheadlongpl</code>	194
<code>\glxtrdiscardperiod</code>	135	<code>\glxtrheadlongpl</code>	194
<code>\glxtrdisplayendloc</code>	93	<code>\Glsxtrheadplural</code>	194
<code>\glxtrdisplayendloohook</code>	94	<code>\glxtrheadplural</code>	194
<code>\glxtrdisplaysingleloc</code>	93, 94	<code>\Glsxtrheadshort</code>	194
<code>\glxtrdisplaystartloc</code>	93	<code>\glxtrheadshort</code>	194
<code>\glxtrdoautoindexname</code>	58, 59, 126	<code>\Glsxtrheadshortpl</code>	194
<code>\glxtrdopostpunc</code>	164	<code>\glxtrheadshortpl</code>	194
<code>\glxtrdowrglossaryhook</code>	59	<code>\Glsxtrheadtext</code>	194
<code>\GlsXtrEnableEntryCounting</code>	82, 83	<code>\glxtrheadtext</code>	194
<code>\GlsXtrEnableEntryUnitCounting</code>	71	<code>\glxtrtrifcounttrigger</code>	74, 75
<code>\GlsXtrEnableOnTheFly</code>	32, 34, 35	<code>\glxtrtrifemptyglossary</code>	93, 98, 102
<code>\glxtrfieldlistgadd</code>	101	<code>\glxtrtrifindexing</code>	58
<code>\glxtrfieldtitlecase</code>	122, 123, 125	<code>\glxtrtrifinmark</code>	67–69, 193–195
<code>\glxtrfieldtitlecasecs</code>	121	<code>\glxtrtrifnextpunc</code>	137, 138
<code>\glxtrfieldxifinlist</code>	103	<code>\glxtrtrifperiod</code>	136
<code>\glxtrfirstscfont</code>	171–174	<code>\glxtrtrifwasfirstuse</code>	
<code>\glxtrfirstsmfont</code>	175–178	42–45, 50–54, 56, 85,
<code>\GlsXtrFmtDefaultOptions</code>	18	135, 136, 144, 146–152, 164, 186, 188–190
<code>\GlsXtrFmtField</code>	18, 19	<code>\glxtrindexaliased</code>	57, 58
<code>\GlsXtrFormatLocationList</code>	36, 39, 224–226	<code>\Glsxtrinlinefullformat</code>	142, 144,
<code>\GLSxtrfull</code>	12, 202, 203	156, 163, 165, 166, 168, 169, 187, 190, 209
<code>\Glsxtrfull</code>	12, 203	<code>\glxtrinlinefullformat</code>	
<code>\glxtrfull</code>	12, 202	142–144, 156, 163, 165–169, 187, 189, 208
<code>\Glsxtrfullformat</code>	142, 154, 156, 159, 161,	<code>\Glsxtrinlinefullplformat</code> ..	142, 145,
.....	163, 165, 167, 168, 170, 186, 187, 189, 192	156, 163, 165, 166, 168, 169, 188, 190, 209
<code>\glxtrfullformat</code> ..	142, 154, 156, 159–	<code>\glxtrinlinefullplformat</code> ..	142, 145, 146,
.....	161, 163, 165, 167–169, 186, 187, 189, 191	156, 163, 165, 166, 168, 169, 187, 189, 209
<code>\GLSxtrfullpl</code>	12, 202, 203	<code>\glxtrinsertinsidefalse</code>	158
<code>\Glsxtrfullpl</code>	12, 203	<code>\glxtrlocrangefmt</code>	94
<code>\glxtrfullpl</code>	12, 202, 203	<code>\GLSxtrlong</code>	12, 200, 201
<code>\Glsxtrfullplformat</code>		<code>\Glsxtrlong</code>	12, 201
.....	142, 154, 156, 159, 161,	<code>\glxtrlong</code>	12, 200
.....	163, 165, 167, 168, 170, 186, 187, 189, 192	<code>\GLSxtrlongpl</code>	12, 201
<code>\glxtrfullplformat</code> .	154, 156, 159, 161,	<code>\Glsxtrlongpl</code>	12, 202
.....	163, 165, 167, 168, 170, 186, 187, 189, 191	<code>\glxtrlongpl</code>	12, 201
<code>\glxtrfullsep</code>	141, 142,	<code>\glxtrlongshortdescsort</code>	159
.....	158–161, 163–166, 168, 169, 179, 180, 184	<code>\glxtrmarkhook</code>	193
<code>\glxtrgenabbrvfmt</code>	39	<code>\glxtrnewabbrvprehook</code>	140
<code>\Glsxtrheadfirst</code>	194	<code>\glxtrnewnumber</code>	12
<code>\glxtrheadfirst</code>	194	<code>\glxtrnewsymbol</code>	12
<code>\Glsxtrheadfirstplural</code>	194	<code>\glxtrNoGlossaryWarning</code>	14, 94
<code>\glxtrheadfirstplural</code>	194	<code>\GlsXtrNoGlsWarningAutoMake</code>	98
<code>\Glsxtrheadfull</code>	194	<code>\GlsXtrNoGlsWarningBuildInfo</code>	98
<code>\glxtrheadfull</code>	194	<code>\GlsXtrNoGlsWarningCheckFile</code>	98

<code>\GlsXtrNoGlsWarningEmptyMain</code>	98	<code>\glxtrrmsuffix</code>	175–178
<code>\GlsXtrNoGlsWarningEmptyNotMain</code>	98	<code>\glxtrrtagfont</code>	133
<code>\GlsXtrNoGlsWarningEmptyStart</code>	98	<code>\Glsxtrrtitlefirst</code>	194, 195, 206
<code>\GlsXtrNoGlsWarningHead</code>	98	<code>\glxtrrtitlefirst</code>	194, 195, 206
<code>\GlsXtrNoGlsWarningMisMatch</code>	98	<code>\Glsxtrrtitlefirstplural</code>	194, 195, 207
<code>\GlsXtrNoGlsWarningNoOut</code>	99	<code>\glxtrrtitlefirstplural</code>	194, 195, 207
<code>\GlsXtrNoGlsWarningTail</code>	99	<code>\Glsxtrrtitlefull</code>	194, 195, 209
<code>\glxtrorg@ifKV@glslink@hyper</code>	40	<code>\glxtrrtitlefull</code>	194, 195, 208
<code>\GLSxtrp</code>	66	<code>\Glsxtrrtitlefullpl</code>	194, 195, 209
<code>\Glsxtrp</code>	66	<code>\glxtrrtitlefullpl</code>	194, 195, 209
<code>\glxtrp</code>	65, 67, 68	<code>\Glsxtrrtitlelong</code>	194, 195, 207
<code>\Glsxtrpl</code>	34	<code>\glxtrrtitlelong</code>	194, 195, 207
<code>\glxtrpl</code>	34	<code>\Glsxtrrtitlelongpl</code>	194, 195, 208
<code>\glxtrpostdescription</code>	119, 134, 217	<code>\glxtrrtitlelongpl</code>	194, 195, 208
<code>\glxtrpostlink</code>	135	<code>\Glsxtrrtitleplural</code>	194, 195, 206
<code>\glxtrpostlinkendsentence</code>	135	<code>\glxtrrtitleplural</code>	194, 195, 205
<code>\glxtrpostlinkhook</code>	134	<code>\Glsxtrrtitleshort</code>	194, 195, 204
<code>\glxtrpostlocalreset</code>	70, 72, 80	<code>\glxtrrtitleshort</code>	194, 195, 204
<code>\glxtrpostlocalunset</code>	70, 72, 80	<code>\Glsxtrrtitleshortpl</code>	194, 195, 205
<code>\glxtrpostlongdescription</code>	23	<code>\glxtrrtitleshortpl</code>	194, 195, 204
<code>\glxtrpostnamehook</code>	124–126	<code>\Glsxtrrtitletext</code>	194, 195, 205
<code>\GlsXtrPostNewAbbreviation</code>		<code>\glxtrrtitletext</code>	194, 195, 205
	141, 156, 158, 160–162, 164, 166, 167, 169, 170, 179, 181, 182, 185, 186, 188–191	<code>\glxtrrtreetopindent</code>	218, 226
<code>\glxtrpostreset</code>	70, 72, 80	<code>\glxtrundefaction</code>	6, 9, 17, 24, 25, 27, 28
<code>\glxtrpostunset</code>	70, 72, 80	<code>\glxtrundeftag</code>	16
<code>\glxtrprotectlinks</code>	60, 61	<code>\glxtrunsrtdo</code>	102, 103
<code>\GlsXtrRecordCounter</code>	7	<code>\GlsXtrUseAbbrStyleFmts</code>	
<code>\glxtrregularfont</code>	39, 42		160, 162, 170–184, 188, 190–192
<code>\glxtrresourcecount</code>	99	<code>\GlsXtrUseAbbrStyleSetup</code>	171–184, 191, 192
<code>\glxtrresourcefile</code>	99	<code>\glxtruserfield</code>	184
<code>\glxtrrestoremarkhook</code>	193	<code>\glxtruserparen</code>	185–192
<code>\glxtrscfont</code>	171–174	<code>\glxtrusersuffix</code>	186, 187, 189, 191
<code>\glxtrscsuffix</code>	171–174	<code>\glxtruseseeformat</code>	29
<code>\GlsXtrSetActualChar</code>	130	<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	138, 157
<code>\glxtrsetaliasnoindex</code>	9, 57, 58	<code>\GlsXtrWarning</code>	32–34
<code>\GlsXtrSetEncapChar</code>	130		
<code>\GlsXtrSetEscChar</code>	130		
<code>\glxtrsetfieldifexists</code>	20, 21		
<code>\GlsXtrSetLevelChar</code>	130		
<code>\glxtrsetpopts</code>	65		
<code>\glxtrsetupfulldefs</code>	143–146, 164		
<code>\GLSxtrshort</code>	12, 69, 195, 196		
<code>\Glsxtrshort</code>	12, 196		
<code>\glxtrshort</code>	12, 195		
<code>\GLSxtrshortpl</code>	12, 196, 197		
<code>\Glsxtrshortpl</code>	12, 197		
<code>\glxtrshortpl</code>	12, 196		
<code>\glxtrrmsfont</code>	174–178		

H

<code>\hangindent</code>	218, 226
<code>\hbox</code>	212
<code>\hfill</code>	212
<code>\href</code>	60
<code>\hsize</code>	35, 36
<code>\hss</code>	212
<code>\hyperlink</code>	60, 61
<code>\hyperpage</code>	127
<code>\hyperref</code>	60
<code>hyperref package</code>	61, 127, 192, 203

I

<code>\if</code>	32
------------------	----

<code>\ifglstr@format@override</code>	127	<code>\ifKV@glslink@noindex</code>	6, 7, 57, 58
<code>\ifglstrdocdefrestricted</code>	30	<code>\ifnum</code>	10, 73, 81, 82, 99, 227
<code>\ifglstrindexcrossrefs</code>	10, 29	<code>\ifthenelse</code>	98
<code>\ifblank</code>	14, 17, 32–34, 86	<code>\IfTrackedLanguageFileExists</code>	210
<code>\ifcase</code>	6, 8, 13, 14, 31, 90	<code>\ifundef</code>	61, 86, 132–134
<code>\ifcsdef</code>	6, 17, 22, 24–27, 54, 55, 65–69, 77, 89, 92, 93, 121–126, 135, 138, 153, 156, 212–216	<code>\ifx</code>	35, 36, 89, 90, 93, 94, 128, 129, 131, 137, 139, 228
<code>\ifcsstring</code>	17, 117, 155	<code>\immediate</code>	73, 81, 82, 95
<code>\ifcsundef</code> ...	22, 24–26, 30, 35, 37, 61, 72, 77–81, 94, 103, 117, 155–157, 211, 219, 226	<code>\index</code>	128
<code>\ifcsvoid</code>	22, 116	<code>\indexspace</code>	228
<code>\ifdef</code>	12, 18, 27, 28, 35, 36, 56, 57, 66, 68, 69, 91, 92, 100, 119, 120, 130, 133, 184, 203–209, 212, 217	<code>\input</code>	209
<code>\ifdefempty</code>	6, 24–26, 29, 71, 83, 86, 89, 102, 104, 132, 153	<code>\inputencodingname</code>	100
<code>\ifdefequal</code>	98, 104	<code>\istfilename</code>	86
<code>\ifdefstring</code>	5, 94, 127, 128, 132	<code>\item</code>	96, 97, 212
<code>\ifdefvoid</code>	22, 28, 30, 60, 77, 103, 104		
<code>\ifdim</code>	35, 36, 85, 219–226		
<code>\IfFileExists</code>	14, 94, 98, 99, 211		
<code>\ifglossaryexists</code>	28		
<code>\ifglssacronym</code>	11, 98		
<code>\ifglssacrshortcuts</code>	13		
<code>\ifglssautomake</code>	89, 98		
<code>\ifglssentrycounter</code>	21		
<code>\ifglssentryexists</code> 27, 28, 32–34, 36, 37, 42, 116, 117, 134		
<code>\ifglssfieldeq</code>	115		
<code>\ifglsshasfield</code>	18, 19, 57, 184		
<code>\ifglsshaslong</code>	76, 77		
<code>\ifglsshasparent</code>	103, 219–222		
<code>\ifglsshasshort</code>	39, 42		
<code>\ifglsshasymbol</code>	135, 218		
<code>\ifglssindexonlyfirst</code>	58		
<code>\ifglssnogroupskip</code>	213–217, 228		
<code>\ifglssnonumberlist</code>	39		
<code>\ifglsssanitizesort</code>	88		
<code>\ifglsssubentrycounter</code>	21		
<code>\ifglssused</code>	29, 30, 56, 58, 73, 82, 85, 153, 219, 220, 222, 224, 225		
<code>\ifglssxindy</code>	94, 96, 97		
<code>\ifglssxtrinsertinside</code> 146–152, 159, 161, 163–170, 186–192		
<code>\ifHy@hyperindex</code>	127		
<code>\ifinlistcs</code>	20, 30		
<code>\ifKV@glslink@hyper</code>	40		
<code>\ifKV@glslink@local</code>	41		
		J	
		<code>\jobname</code>	94, 96–99
		K	
		<code>\key@ifundefined</code>	8, 17, 18, 54, 102, 104
		<code>\KV@glslink@hyperfalse</code>	43, 56, 61
		<code>\KV@glslink@noindexfalse</code>	57
		<code>\KV@glslink@noindextrue</code>	57, 61
		L	
		<code>\LaTeX</code>	96, 97
		<code>\leaders</code>	212
		<code>\leavevmode</code>	24
		<code>\let</code> ..	4, 6, 7, 9, 11–13, 15, 16, 18, 23, 30, 31, 35, 37, 38, 40–62, 65, 71–73, 80–87, 89– 92, 99, 101, 102, 104, 121–129, 132–134, 137–140, 143–152, 164, 193–195, 217, 219
		<code>\letabbreviationstyle</code>	164, 165, 167, 168, 170, 172, 173, 176, 181, 182
		<code>\letcs</code> ..	17, 29, 30, 54, 91, 92, 103, 104, 121–126
		<code>\levelchar</code>	130
		<code>\listadd</code>	77
		<code>\listbreak</code>	132
		<code>\listcsadd</code>	19
		<code>\listcseadd</code>	19, 78
		<code>\listcsgadd</code>	19, 30, 31
		<code>\listcsxadd</code>	19, 78
		<code>\loadglssentries</code>	31, 96
		<code>\long</code>	23
		M	
		<code>\MakeAcronymsAbbreviations</code>	85
		<code>\makeatletter</code>	94, 99, 129
		<code>\makeatother</code>	129
		<code>\makebox</code>	212, 227

<code>\makefirsttuc</code>	133	<code>firstplural</code> ..	107, 113, 158, 199, 200, 206, 229
<code>makeglossaries</code>	90	<code>loclist</code>	19
<code>\makeglossaries</code>	86, 95, 97, 98	<code>long</code>	111, 115, 207
<code>\makeglossary</code>	86	<code>longplural</code>	112, 115, 208
<code>makeindex</code>	229	<code>name</code>	105, 112, 127
<code>makeindex</code>	86	<code>plural</code>	106, 112, 113, 158, 198, 205
<code>\makenoidxglossaries</code>	97	<code>see</code>	10, 29, 31, 86
<code>\MakeTextUppercase</code>	194	<code>short</code>	110, 114, 139
<code>\MakeUppercase</code>	194, 195	<code>shortplural</code>	111, 114, 139
<code>\markboth</code>	193	<code>symbol</code>	108, 113
<code>\markright</code>	193	<code>symbolplural</code>	108, 113
<code>\maxdimen</code>	35, 36	<code>text</code> ..	60, 105, 106, 112, 158, 159, 197, 198, 205
<code>\mbox</code>	227	<code>\newif</code>	127, 158
<code>\medskip</code>	98, 103	<code>\newlength</code>	218
<code>\MessageBreak</code>	31, 34, 73, 82, 88, 89, 155	<code>\newnum</code>	12
<code>mfistuc</code> package	132, 133	<code>\newrobustcmd</code>	18–21, 55, 65, 66, 76, 132, 133, 143–152, 193, 195–203, 219–225
<code>\mfistucMakeUppercase</code>	43–55, 63–66, 69, 76, 84, 105–115, 124, 125, 144, 146, 147, 149, 151–154	<code>\newsym</code>	12
<code>\mfu@checkword@arg</code>	132, 133	<code>\newterm</code>	119
<code>\mfu@checkword@do</code>	133	<code>\newtoks</code>	138, 139
N		<code>\newwrite</code>	86
<code>\NeedsTeXFormat</code>	4, 211	<code>\NoCaseChange</code>	67–69, 195–203
<code>\new@glossaryentry</code>	31, 88	<code>\noexpand</code>	7, 14, 22, 83, 84, 95, 99, 128, 129, 141, 211
<code>\new@ifnextchar</code>	54, 55, 76, 136, 143–152	<code>\nofiles</code>	97
<code>\newabbr</code>	12	<code>\noindent</code>	98
<code>\newabbreviation</code>	12	<code>\nopostdesc</code>	24, 33, 34, 90, 119
<code>\newabbreviationhook</code>	141	<code>\nr</code>	6, 8, 10, 13, 14, 90
<code>\newabbreviationstyle</code>	158–162, 164, 166, 167, 169–186, 188, 190–192	<code>\ns@GLSxtrfull</code>	144
<code>\newacronym</code>	83, 84	<code>\ns@Glsxtrfull</code>	144
<code>\newacronymhook</code>	83	<code>\ns@glxtrfull</code>	143
<code>\newacronymstyle</code>	84, 85	<code>\ns@GLSxtrfullpl</code>	146
<code>\newcommand</code>	4–8, 10–35, 37–39, 42, 43, 54, 55, 57–61, 65–73, 76–86, 89–92, 94–103, 105–121, 126–139, 141–153, 155–157, 159, 162, 170, 171, 174, 175, 178, 184, 185, 193, 195–211, 217–219, 226	<code>\ns@Glsxtrfullpl</code>	145
<code>\newcount</code>	9, 99	<code>\ns@glxtrfullpl</code>	145
<code>\newentry</code>	12	<code>\ns@GLSxtrlong</code>	149
<code>\newglossary</code>	11, 86	<code>\ns@Glsxtrlong</code>	148
<code>\newglossaryentry</code>	12, 31, 71, 72, 79, 83, 119, 120, 141	<code>\ns@glxtrlong</code>	148
<code>\newglossaryentry options</code>	22	<code>\ns@GLSxtrlongpl</code>	152
<code>alias</code>	109, 114	<code>\ns@Glsxtrlongpl</code>	151, 152
<code>desc</code>	109, 110, 114	<code>\ns@glxtrlongpl</code>	151
<code>descplural</code>	109, 110, 114	<code>\ns@GLSxtrshort</code>	147
<code>first</code> ..	60, 106, 107, 113, 158, 198–200, 206, 229	<code>\ns@Glsxtrshort</code>	147
		<code>\ns@glxtrshort</code>	146
		<code>\ns@GLSxtrshortpl</code>	150
		<code>\ns@Glsxtrshortpl</code>	150
		<code>\ns@glxtrshortpl</code>	149
		<code>\null</code>	14
		<code>\number</code>	78–81, 99
		<code>\numexpr</code>	78, 79, 81

O	
<code>\or</code>	6, 9, 13, 31
<code>\org@glossaryentrynumbers</code>	36, 90
<code>\org@glossarytitle</code>	89, 90
P	
<code>\p@glshyp@opt</code>	59
package options:	
abbreviations	11
accsupp	14, 104
acronym	11
automake	89, 96
docdef	9, 30, 31, 71, 79
false	31
restricted	10
true	31
nonumberlist	36
nopostdot	
false	11
numbers	12
record	6, 8, 39, 99, 238
shortcuts	13
all	13
false	13
none	13
true	13
style	15
stylemods	15
symbols	12, 119
undefaction	27
error	5
warn	5
<code>\PackageError</code>	5, 7, 15, 31, 34, 35, 54–56, 58, 65, 66, 71–73, 79, 81, 83, 84, 86, 88, 93, 103, 155–157, 211
<code>\PackageWarning</code>	10
<code>\PackageWarningNoLine</code>	11
<code>\pageref</code>	21
<code>\par</code>	98, 99, 217, 218, 227
<code>\parindent</code>	218, 227
<code>\PassOptionsToPackage</code>	4
<code>\preglossarypreamble</code>	22
<code>\preto</code>	57
<code>\print@noop@unsrtglossaryunit</code>	7, 9
<code>\print@op@unsrtglossaryunit</code>	9
<code>\printabbreviations</code>	11
<code>\printglossaries</code>	87, 97
<code>\printglossary</code>	11, 87, 97
<code>\printglossary options</code>	
nonumberlist	38
type	89
<code>\printnoidxglossaries</code>	97
<code>\printnoidxglossary</code>	87, 97
<code>\printnumbers</code>	12, 120
<code>\printsymbols</code>	12, 119
<code>\printunsrtglossary</code>	102
<code>\printunsrtglossaryhandler</code>	102, 103
<code>\printunsrtglossaryunit</code>	9, 103
<code>\printunsrtglossaryunitsetup</code>	102
<code>\ProcessOptions</code>	211
<code>\ProcessOptionsX</code>	15
<code>\protect</code>	67–69, 112– 115, 141, 142, 158–164, 166, 167, 169– 180, 182, 185, 186, 188, 190, 191, 195–203
<code>\protected@csdef</code>	21, 218
<code>\protected@csxdef</code>	21, 218
<code>\protected@edef</code>	35, 83, 127, 141
<code>\protected@write</code>	7, 38, 86, 87, 99–101
<code>\protected@xdef</code>	104
<code>\providecommand</code>	11, 18, 38, 56, 57, 73, 82, 86, 95, 211
<code>\ProvidesFile</code>	210
<code>\ProvidesPackage</code>	4, 211
Q	
<code>\quotechar</code>	130
R	
<code>\raggedright</code>	214, 216
<code>\relax</code>	6, 8–10, 12– 16, 18, 31, 35, 36, 38, 41, 59, 65, 73, 81, 82, 86, 87, 89, 90, 92, 93, 99, 101, 103, 128, 129, 131, 133, 135, 139, 140, 219–228
<code>resize package</code>	174
<code>\renewcommand</code>	5, 9–11, 13–16, 23–28, 30–32, 34–39, 41, 54, 56–58, 60, 61, 65, 70–73, 76, 77, 79–90, 92–95, 103, 119, 121–126, 131– 134, 142, 156, 158–193, 212–217, 227, 228
<code>\renewenvironment</code>	212–216, 226
<code>\renewglossarystyle</code>	212–216, 226
<code>\renewrobustcmd</code>	42
<code>\RequireGlossariesExtraLang</code>	210
<code>\RequirePackage</code>	4, 14, 15, 211
<code>\reserved@a</code>	137
<code>\reserved@b</code>	137
<code>\reserved@d</code>	137
<code>\RestoreAcronyms</code>	85
<code>\romannumeral</code>	218, 219, 226

S		<code>\texttt</code> 95–98
<code>\s@glshyp@opt</code> 59	<code>\the</code> 83, 84, 99, 131, 141, 158–162, 164, 166, 167, 169, 170, 179–182, 185–191	<code>\theindex</code> 127
<code>\s@glstr@enabletagging</code> 132	<code>\this@dialect</code> 210	<code>\toks@</code> 131
<code>\s@printunsrtglossary</code> 101, 102	<code>\tracklang package</code> 100	
<code>\seename</code> 29	U	
<code>\setabbreviationstyle</code> 85, 159, 167	<code>\undef</code> 9, 132	<code>\underline</code> 133
<code>\setacronymstyle</code> 84, 85	<code>\unskip</code> 24, 30, 212	<code>\usepackage</code> 97, 98
<code>\setentrycounter</code> 93	V	
<code>\SetGenericNewAcronym</code> 85	<code>\val</code> 6, 8, 10, 13, 14, 90	
<code>\setglossarystyle</code> 15, 89, 212, 228	W	
<code>\setkeys</code> 6, 15, 16, 42, 58, 83, 90, 139, 140	<code>\warn@nomakeglossaries</code> 87	<code>\warn@noprintglossary</code> 87, 90
<code>\setlength</code> 35, 36, 218, 227	<code>\write</code> 73, 81, 82, 86, 95	
<code>\settowidth</code> 85, 218–226	X	
<code>\setupglossaries</code> 4, 16	<code>\xcapitalisewords</code> 121	<code>\xdef</code> 90
<code>\sfcode</code> 135, 217	<code>\xifinlist</code> 77	<code>\xifinlistcs</code> 20
<code>\space</code> 5, 7, 32, 34, 58, 71–73, 79, 81–83, 85–89, 95, 98, 103, 135, 142, 159, 217, 218, 226	<code>xindy</code> 229	<code>xindy</code> 86
<code>\spacefactor</code> 135, 140, 217	<code>xkeyval package</code> 4	<code>\XKV@checkchoice</code> 38
<code>\string</code> 5, 7, 31, 32, 34, 38, 54, 55, 58, 65, 66, 71–73, 79, 81–89, 95–101, 103, 123–126, 128	<code>\XKV@plfalse</code> 38	<code>\XKV@resa</code> 38, 39
<code>\strut</code> 212–217	<code>\XKV@sttrue</code> 38	
<code>\subglossentry</code> ... 90, 103, 104, 212–217, 227		
T		
<code>\tablehead</code> 215, 216		
<code>\tabletail</code> 215, 216		
<code>\tabularnewline</code> 212–217		
<code>\TeX</code> 96		
<code>\texorpdfstring</code> 18, 19, 66–69, 203–209		
<code>textcase package</code> 192		
<code>\textsc</code> 170		
<code>\textsmaller</code> 174		