

glossaries-extra.sty v1.12: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2017-02-03

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	34
1.3.4 Entry Formatting, Hyperlinks and Indexing	39
1.3.5 Entry Counting	69
1.3.6 Acronym Modifications	82
1.3.7 Indexing and Displaying Glossaries	85
1.4 Integration with glossaries-accsupp	104
1.5 Categories	115
1.6 Abbreviations	137
1.6.1 Abbreviation Styles Setup	154
1.6.2 Predefined Styles (Default Font)	157
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}[2017/02/03 v1.12 (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 \glsxtr@@do@@wrglossary.

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

glsxtr@dorecord

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

r@recordcounter

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

p@recordcounter

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

p@recordcounter

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

srtglossaryunit

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

tr@setup@record Initialise.

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

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

```
119 \newcommand*\@glsxtr@indexonly@saveentrycounter{%
120 \ifKV@glslink@noindex
121 \else
122 \glsxtr@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*{\@glxtr@record}[2]{}%
167     \let\@do@wrglossary\glxtr@do@wrglossary
168     \let\@gl@saveentrycounter\glxtr@indexonly@saveentrycounter
169     \let\glxtrundefaction\glxtr@err@undefaction
170     \let\glxtr@warnonexistsordo\@gobble
171     \let\@glxtr@recordcounter\glxtr@noop@recordcounter
172     \def\printunsrtglossaryunit{\print@noop@unsrtglossaryunit}%
173 }%
174 \or

```

Only record (don't index).

```

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

```

Record and index.

```

187     \def\glxtr@setup@record{%
188     \let\@glxtr@record\@glxtr@record
189     \let\@do@wrglossary\glxtr@do@wrglossary
190     \let\@gl@saveentrycounter\glxtr@indexonly@saveentrycounter
191     \let\glxtrundefaction\glxtr@warn@undefaction
192     \let\glxtr@warnonexistsordo\glxtr@warn@onexistsordo
193     \glxtr@addloclistfield
194     \let\@glxtr@recordcounter\glxtr@op@recordcounter
195     \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
196 }%
197 \fi
198 }

```

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

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

```

199 \newcount\@glxtr@docdefval

```

Need to provide conditional commands that are backward compatible:

```

if@glxtrdocdef

```

```

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

```

lsxtrdocdeftrue

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

lsxtrdocdeffalse

```
202 \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.

```
203 \define@choicekey{glossaries-extra.sty}{docdef}[\val\nr]%  
204 {false,true,restricted}[true]%  
205 {%  
206   \@glxtr@docdefval=\nr\relax  
207   \ifnum\@glxtr@docdefval=2\relax  
208     \renewcommand*{\@glsdoidexistsorwarn}{\glsdoidexists}%  
209   \fi  
210 }
```

ocdefrestricted

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

oidexistsorwarn

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).

```
212 \newcommand*{\@glsdoidexistsorwarn}{\glsdoidexistsorwarn}
```

indexcrossrefs

Automatically index cross references at the end of the document

```
213 \define@boolkey{glossaries-extra.sty}[@glxtr]{indexcrossrefs}[true]{%  
214   \if@glxtrindexcrossrefs  
215   \else  
216     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%  
217   \fi  
218 }
```

Switch off since this can increase the build time.

```
219 \@glxtrindexcrossrefsfalse
```

But allow see key to switch it on automatically.

oindexcrossrefs

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

iesExtraWarning

Allow users to suppress warnings.

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

raWarningNoLine

Allow users to suppress warnings.

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

```

224 \@glxtr@declareoption{nowarn}{%
225   \let\GlossariesExtraWarning\@gobble
226   \let\GlossariesExtraWarningNoLine\@gobble
227   \glxtr@doption{nowarn}%
228 }

postdot   Shortcut for nopostdot=false
229 \@glxtr@declareoption{postdot}{%
230   \glxtr@doption{nopostdot=false}%
231 }

glxtrabbrvtype   Glossary type for abbreviations.
232 \newcommand*{\glxtrabbrvtype}{\glsdefaulttype}

abbreviationsdef   Set by abbreviations option.
233 \newcommand*{\@glxtr@abbreviationsdef}{}

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

   If the acronym option hasn't been used, change \acronymtype to \glxtrabbrvtype.
244   \ifglsacronym
245   \else
246     \renewcommand*{\acronymtype}{\glxtrabbrvtype}%
247   \fi
248 }%

abbreviations   If abbreviations, create a new glossary type for abbreviations.
249 \@glxtr@declareoption{abbreviations}{%
250   \let\@glxtr@abbreviationsdef\@glxtr@doabbreviationsdef
251 }

abbreviationShortcuts   Enable shortcut commands for the abbreviations. Unlike the analogous command provided
                        by glossaries, this uses \newcommand instead of \let as a safety feature.
252 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
253   \newcommand*{\ab}{\cgl}%
254   \newcommand*{\abp}{\cglsp}%
255   \newcommand*{\as}{\glxtrshort}%
256   \newcommand*{\asp}{\glxtrshortpl}%
257   \newcommand*{\al}{\glxtrlong}%

```

```

258 \newcommand*\alp{\glstrlongpl}%
259 \newcommand*\af{\glstrfull}%
260 \newcommand*\afp{\glstrfullpl}%
261 \newcommand*\Ab{\cGls}%
262 \newcommand*\Abp{\cGlspl}%
263 \newcommand*\As{\Glsxtrshort}%
264 \newcommand*\Asp{\Glsxtrshortpl}%
265 \newcommand*\Al{\Glsxtrlong}%
266 \newcommand*\Alp{\Glsxtrlongpl}%
267 \newcommand*\Af{\Glsxtrfull}%
268 \newcommand*\Afp{\Glsxtrfullpl}%
269 \newcommand*\AB{\cGLS}%
270 \newcommand*\ABP{\cGLSpl}%
271 \newcommand*\AS{\GLSxtrshort}%
272 \newcommand*\ASP{\GLSxtrshortpl}%
273 \newcommand*\AL{\GLSxtrlong}%
274 \newcommand*\ALP{\GLSxtrlongpl}%
275 \newcommand*\AF{\GLSxtrfull}%
276 \newcommand*\AFP{\GLSxtrfullpl}%
277 \newcommand*\newabbr{\newabbreviation}%

Disable this command after it's been used.
278 \let\GlsXtrDefineAbbreviationShortcuts\relax
279 }

```

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

```

280 \newcommand*\GlsXtrDefineOtherShortcuts{%
281 \newcommand*\newentry{\newglossaryentry}%
282 \ifdef\printsymbols
283 {%
284 \newcommand*\newsym{\glstrnewsymbol}%
285 }{}%
286 \ifdef\printnumbers
287 {%
288 \newcommand*\newnum{\glstrnewnumber}%
289 }{}%
290 \let\GlsXtrDefineOtherShortcuts\relax
291 }

```

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.

```

292 \newcommand*\@glstr@setupshortcuts{}

```

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

```

293 \newcommand*\@glstr@shortcutsval{\ifglsacrshortcutsacro\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.

```

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

```

lsxtr@doaccsupp

```
337 \newcommand*{\@glxtr@doaccsupp}{}
```

accsupp If accsupp, load glossaries-accsupp package.

```
338 \@glxtr@declareoption{accsupp}{%
```

```
339 \renewcommand*{\@glxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}
```

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

```
340 \newcommand{\glxtrNoGlossaryWarning}[1]{%
```

```
341 \@glxtr@defaultnoglossarywarning{#1}%
```

```
342 }
```

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

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

```
344 {true,false}[true]{%
```

```
345 \ifcase\nr\relax % true
```

```
346 \renewcommand{\glxtrNoGlossaryWarning}[1]{%
```

```
347 \null
```

```
348 }%
```

```
349 \else % false
```

```
350 \renewcommand{\glxtrNoGlossaryWarning}[1]{%
```

```
351 \@glxtr@defaultnoglossarywarning{#1}%
```

```
352 }%
```

```
353 \fi
```

```
354 }
```

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

lxtr@redefstyles

```
355 \newcommand*{\@glxtr@redefstyles}{}
```

stylemods

```
356 \define@key{glossaries-extra.sty}{stylemods}{%
```

```
357 \ifblank{#1}{%
```

```
358 {%
```

```
359 \renewcommand*{\@glxtr@redefstyles}{%
```

```
360 \RequirePackage{glossaries-extra-stylemods}}%
```

```
361 }%
```

```
362 {%
```

```
363 \renewcommand*{\@glxtr@redefstyles}{}%
```

```
364 \@for\@glxtr@tmp:=#1\do{%
```

```
365 \IfFileExists{glossary-\@glxtr@tmp.sty}{%
```

```
366 {%
```

```
367 \eappto\@glxtr@redefstyles{%
```

```
368 \noexpand\RequirePackage{glossary-\@glxtr@tmp}}%
```

```
369 }%
```

```
370 {%
```

```
371 \PackageError{glossaries-extra}{%
```

```

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

```

glxtr@do@style

```

382 \newcommand*{\@glxtr@do@style}{}

```

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

```

383 \define@key{glossaries-extra.sty}{style}{%
384 \renewcommand*{\@glxtr@do@style}{%

```

Set this as the default style:

```

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

```

Set this style:

```

386 \setglossarystyle{#1}%
387 }%
388 }

```

Pass all other options to glossaries.

```

389 \DeclareOptionX*{%
390 \expandafter\glxtr@doooption\expandafter{\CurrentOption}}

```

Process options.

```

391 \ProcessOptionsX

```

Load glossaries if not already loaded.

```

392 \RequirePackage{glossaries}

```

Load the glossaries-accsupp package if required.

```

393 \@glxtr@doaccsupp

```

Define abbreviations glossaries if required.

```

394 \@glxtr@abbreviationsdef
395 \let\@glxtr@abbreviationsdef\relax

```

Setup shortcuts if required.

```

396 \@glxtr@setupshortcuts

```

Redefine \@glxtr@redef@forlglentries if required.

```

397 \@glxtr@redef@forlglentries

```

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

```

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

```

Now define the user command:

```

399 \newcommand*{\glossariesextrasetup}[1]{%
400   \let\glsxtr@setup@record\relax
401   \let\glsxtr@setupshortcuts\relax
402   \let\glsxtr@redef@for@gl@sentries\relax
403   \setkeys{glossaries-extra.sty}{#1}%
404   \glsxtr@abbreviationsdef
405   \let\glsxtr@abbreviationsdef\relax
406   \glsxtr@setupshortcuts
407   \glsxtr@setup@record
408   \glsxtr@redef@for@gl@sentries
409 }
```

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

```
410 \let\glsxtr@@do@wrglossary\@@do@wrglossary
```

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

```
411 \let\glsxtr@saveentrycounter\@gls@saveentrycounter
```

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

```
412 \let\@gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
```

Set up record option if required.

```
413 \glsxtr@setup@record
```

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

```

414 \AtBeginDocument{%
415   \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
416   \def\@glsxtrundeftag{\glsxtrundeftag}%
417 }
```

1.2 Extra Utilities

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

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

```

418 \newcommand{\glsxtrifemptyglossary}[3]{%
419   \ifcsdef{glolist@#1}%
```



```

420 {%
421   \ifcsstring{glolist@#1}{,}{#2}{#3}%
422 }%
423 {%
424   \glstrundefaction{Glossary type ‘#1’ doesn’t exist}{}%
425   #2%
426 }%
427 }

```

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

```

428 \newcommand*{\glxtrifkeydefined}[3]{%
429   \key@ifundefined{glossentry}{#1}{#3}{#2}%
430 }

```

providestoragekey Like `\glsaddstoragekey` but does nothing if the key has already been defined.

```

431 \newcommand*{\glxtrprovidestoragekey}{%
432   \@ifstar\sglsxtr@provide@storagekey\glxtr@provide@storagekey
433 }

```

provide@storagekey Unstarred version.

```

434 \newcommand*{\@glxtr@provide@storagekey}[3]{%
435   \key@ifundefined{glossentry}{#1}%
436   {%
437     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
438     \appto\@gls@keymap{,}{#1}{#1}}%
439     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
440     \appto\@newglossaryentryposthook{%
441       \letcs{@glo@tmp}{@glo@#1}%
442       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
443     }%

```

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

```

444   \ifblank{#3}
445   {%
446     {%
447       \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
448     }%
449   }%
450   {%

```

Provide the no-link command if not already defined.

```

451   \ifblank{#3}
452   {%
453     {%
454       \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
455     }%
456   }%
457 }

```

vide@storagekey Starred version.

```
458 \newcommand*{\s@glxstr@provide@storagekey}[1]{%
459   \key@ifundefined{glossentry}{#1}%
460   {%
461     \expandafter\newcommand\expandafter*\expandafter
462     {\csname gls@assign@#1@field@endcsname}[2]{%
463       \@@gls@expand@field{##1}{#1}{##2}%
464     }%
465   }%
466   {%
467     \@glxstr@provide@addstoragekey{#1}%
468   }
```

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

```
469 \newcommand{\GlsXtrFmtField}{useri}
```

`tDefaultOptions`

```
470 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}
```

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

```
471 \newrobustcmd*{\glsxtrfmt}[3][[]]{%
472   \glsdoifexistsordo{#2}%
473   {%
474     \ifglshasfield{\GlsXtrFmtField}{#2}%
475     {%
476       \let\do@gls@link@checkfirsthyper\relax
477       \expandafter\@gls@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
478       {\csuse{\glscurrentfieldvalue}{#3}}%
479     }%
480     {#3}%
481   }%
482   {#3}%
483 }
```

`\glsxtrentryfmt` No link or indexing.

```
484 \ifdef\texorpdfstring
485 {
486   \newcommand*{\glsxtrentryfmt}[2]{%
487     \texorpdfstring{\@glsxtrentryfmt{#1}{#2}}{#2}%
488   }
489 }
490 {
491   \newcommand*{\glsxtrentryfmt}{\@glsxtrentryfmt}
492 }
```

@glxxtrententryfmt

```
493 \newrobustcmd*{\@glxxtrententryfmt}[2]{%
494 \glsoifexistsordo
495 {%
496 \ifglshasfield{\GlsXtrFmtField}{#1}%
497 {%
498 \csuse{\glscurrentfieldvalue}{#2}%
499 }%
500 {#2}%
501 }%
502 {#2}%
503 }
```

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.

```
504 \newcommand*{\glxtrfieldlistadd}[3]{%
505 \listcsadd{glo@\glsetoklabel{#1}@#2}{#3}%
506 }
```

trfieldlistgadd Similarly but uses \listcsgadd.

```
507 \newcommand*{\glxtrfieldlistgadd}[3]{%
508 \listcsgadd{glo@\glsetoklabel{#1}@#2}{#3}%
509 }
```

trfieldlisteadd Similarly but uses \listcseadd.

```
510 \newcommand*{\glxtrfieldlisteadd}[3]{%
511 \listcseadd{glo@\glsetoklabel{#1}@#2}{#3}%
512 }
```

trfieldlistxadd Similarly but uses \listcsxadd.

```
513 \newcommand*{\glxtrfieldlistxadd}[3]{%
514 \listcsxadd{glo@\glsetoklabel{#1}@#2}{#3}%
515 }
```

Now provide commands to iterate over these lists.

fielddolistloop

```
516 \newcommand*{\glxtrfielddolistloop}[2]{%
517 \dolistcsloop{glo@\glsetoklabel{#1}@#2}%
518 }
```

ieldforlistloop

```
519 \newcommand*{\glxtrfieldforlistloop}[3]{%
520 \forlistcsloop{glo@\glsetoklabel{#1}@#2}{#3}%
521 }
```

List element tests:

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

```

522 \newcommand*{\glstrfieldifinlist}[5]{%
523   \ifinlistcs{#3}{glo@\glstetoklabel{#1}@#2}{#4}{#5}%
524 }

```

rfieldxifinlist Expands item.

```

525 \newcommand*{\glstrfieldxifinlist}[5]{%
526   \xifinlistcs{#3}{glo@\glstetoklabel{#1}@#2}{#4}{#5}%
527 }

```

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

```

528 \newcommand*{\glstrusefield}[2]{%
529   \@gls@entry@field{#1}{#2}%
530 }

```

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

```

531 \newcommand*{\Glsxtrusefield}[2]{%
532   \@Gls@entry@field{#1}{#2}%
533 }

```

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

```

534 \newcommand*{\glstrdeffield}[2]{\csdef{glo@\glstetoklabel{#1}@#2}}

```

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

```

535 \newcommand*{\glxstredeffield}[2]{\csedef{glo@\glstetoklabel{#1}@#2}}

```

etfieldifexists

```

536 \newcommand*{\glstrsetfieldifexists}[3]{\glstoifexists{#1}{#3}}

```

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

```

537 \newrobustcmd*{\GlsXtrSetField}[3]{%
538   \glstrsetfieldifexists{#1}{#2}%
539   {\csdef{glo@\glstetoklabel{#1}@#2}{#3}}%
540 }

```

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

```

541 \newrobustcmd*{\GlsXtrLetField}[3]{%
542   \glstrsetfieldifexists{#1}{#2}%
543   {\cslet{glo@\glstetoklabel{#1}@#2}{#3}}%
544 }

```

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

```

545 \newrobustcmd*{\csGlsXtrLetField}[3]{%
546   \glstrsetfieldifexists{#1}{#2}%
547   {\csletcs{glo@\glstetoklabel{#1}@#2}{#3}}%
548 }

```

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.

```
549 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
550   \glstrsetfieldifexists{#1}{#2}%
551   {\csletcs{glo@glstetoklabel{#1}@#2}{glo@glstetoklabel{#3}@#4}}%
552 }
```

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

```
553 \newrobustcmd*{\gGlsXtrSetField}[3]{%
554   \glstrsetfieldifexists{#1}{#2}%
555   {\csgdef{glo@glstetoklabel{#1}@#2}{#3}}%
556 }
```

xGlsXtrSetField

```
557 \newrobustcmd*{\xGlsXtrSetField}[3]{%
558   \glstrsetfieldifexists{#1}{#2}%
559   {\protected@csxdef{glo@glstetoklabel{#1}@#2}{#3}}%
560 }
```

eGlsXtrSetField

```
561 \newrobustcmd*{\eGlsXtrSetField}[3]{%
562   \glstrsetfieldifexists{#1}{#2}%
563   {\protected@csedef{glo@glstetoklabel{#1}@#2}{#3}}%
564 }
```

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

```
565 \ifglssentrycounter
566   \newcommand*{\glstrpageref}[1]{\pageref{glssentry-\glstetoklabel{#1}}}
567 \else
568   \ifglssubentrycounter
569     \newcommand*{\glstrpageref}[1]{\pageref{glssentry-\glstetoklabel{#1}}}
570   \else
571     \newcommand*{\glstrpageref}[1]{\glss{#1}}
572   \fi
573 \fi
```

lossarypreamble

```
574 \newcommand{\apptoglossarypreamble}[2][\glssdefaulttype]{%
575   \ifcsdef{glolist@#1}%
576   {%
577     \ifcsundef{@glossarypreamble@#1}%
578     {\csdef{@glossarypreamble@#1}{}}%
579   }%
580   \csappto{@glossarypreamble@#1}{#2}%
581   }%
582   {%
583     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
```

```

584 }%
585 }

```

lossarypreamble

```

586 \newcommand{\preglossarypreamble}[2][\glsdefaultttype]{%
587   \ifcsdef{glolist@#1}%
588   {%
589     \ifcsundef{@glossarypreamble@#1}%
590     {\csdef{@glossarypreamble@#1}{}}%
591     {}%
592     \cspretoto{@glossarypreamble@#1}{#2}%
593   }%
594   {%
595     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
596   }%
597 }

```

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.

```

598 \glsaddstoragekey*{alias}{\glsxtralias}

```

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

```

599 \appto\newglossaryentryposthook{%
600   \ifcsvoid{glo@\glo@label @alias}{}%
601   {%
602     \ifdefvoid\@glo@see
603     {%
604       \edef\@do@glssee{\noexpand\glssee
605         {\@glo@label}{\csuse{glo@\glo@label @alias}}}%
606       \@do@glssee
607     }%
608     {}%
609   }%
610 }

```

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 `\glsxtrpostlongdescription` instead.

ewglossaryentry

```

611 \renewcommand*{\longnewglossaryentry}{%

```

```

612 \@ifstar\@glxtr@s@longnewglossaryentry\@glxtr@longnewglossaryentry
613 }

```

ewglossaryentry Starred version.

```

614 \newcommand{\@glxtr@s@longnewglossaryentry}[3]{%
615   \glsdoifnoexists{#1}%
616   {%
617     \bgroup
618     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
619     \long\def\@newglossaryentryprehook{%
620       \long\def\@glo@desc{#3}%
621       \@org@newglossaryentryprehook
622     }%
623     \renewcommand*\@gls@assign@desc}[1]{%
624       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
625       \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
626     }
627     \gls@defglossaryentry{#1}{#2}%
628   \egroup
629 }%
630 }

```

ewglossaryentry Unstarred version.

```

631 \newcommand{\@glxtr@longnewglossaryentry}[3]{%
632   \glsdoifnoexists{#1}%
633   {%
634     \bgroup
635     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
636     \long\def\@newglossaryentryprehook{%
637       \long\def\@glo@desc{#3\glsxtrpostlongdescription}%
638       \@org@newglossaryentryprehook
639     }%
640     \renewcommand*\@gls@assign@desc}[1]{%
641       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%

```

The following is different from the base glossaries.sty:

```

642       \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
643     }
644     \gls@defglossaryentry{#1}{#2}%
645   \egroup
646 }%
647 }

```

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

```

648 \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.

```
649 \renewcommand{\newignoredglossary}{%
650   \@ifstar\glxtr@s@newignoredglossary\glxtr@org@newignoredglossary
651 }
```

ignoredglossary The original definition is patched to check for existence.

```
652 \newcommand*{\glxtr@org@newignoredglossary}[1]{%
653   \ifcsdef{glolist@#1}
654   {%
655     \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
656   }%
657   {%
658     \ifdefempty\@ignored@glossaries
659     {%
660       \edef\@ignored@glossaries{#1}%
661     }%
662     {%
663       \eappto\@ignored@glossaries{, #1}%
664     }%
665     \csgdef{glolist@#1}{,}%
666     \ifcsundef{gls@#1@entryfmt}%
667     {%
668       \defglsentryfmt[#1]{\glsentryfmt}%
669     }%
670     {}%
671     \ifdefempty\@gls@nohyperlist
672     {%
673       \renewcommand*{\@gls@nohyperlist}{#1}%
674     }%
675     {%
676       \eappto\@gls@nohyperlist{, #1}%
677     }%
678   }%
679 }
```

ignoredglossary Starred form.

```
680 \newcommand*{\glxtr@s@newignoredglossary}[1]{%
681   \ifcsdef{glolist@#1}
682   {%
683     \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
684   }%
685   {%
686     \ifdefempty\@ignored@glossaries
687     {%
688       \edef\@ignored@glossaries{#1}%
689     }%
690     {%
691       \eappto\@ignored@glossaries{, #1}%
692     }%
693   }
```



```

693 \csgdef{glolist@#1}{,}%
694 \ifcsundef{gls@#1@entryfmt}%
695 {%
696 \defglsentryfmt[#1]{\glsentryfmt}%
697 }%
698 {}%
699 }%
700 }

```

`\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.

```

701 \glsifusetranslator
702 {%
703 \renewcommand*{\glsettoctitle}[1]{%
704 \ifcsdef{gls@tr@set@#1@toctitle}%
705 {%
706 \csuse{gls@tr@set@#1@toctitle}%
707 }%
708 {%
709 \ifcsdef{@glotype@#1@title}%
710 {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
711 {\def\glossarytoctitle{\glossarytitle}}%
712 }%
713 }%
714 }
715 {
716 \renewcommand*{\glsettoctitle}[1]{%
717 \ifcsdef{@glotype@#1@title}%
718 {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
719 {\def\glossarytoctitle{\glossarytitle}}%
720 }
721 }

```

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

```

722 \newcommand{\provideignoredglossary}{%
723 \@ifstar\glsxtr@s@provideignoredglossary\glsxtr@provideignoredglossary
724 }

```

`ignoredglossary` Unstarred version.

```

725 \newcommand*{\glsxtr@provideignoredglossary}[1]{%
726 \ifcsdef{glolist@#1}
727 {}%
728 {%
729 \ifdefempty\@ignored@glossaries
730 {%
731 \edef\@ignored@glossaries{#1}%
732 }%
733 {%
734 \eappto\@ignored@glossaries{, #1}%

```

```

735 }%
736 \csgdef{glolist@#1}{,}%
737 \ifcsundef{gls@#1@entryfmt}%
738 {%
739   \defglsentryfmt[#1]{\glsentryfmt}%
740 }%
741 {}%
742 \ifdefempty{@gls@nohyperlist
743 {%
744   \renewcommand*{@gls@nohyperlist}{#1}%
745 }%
746 {%
747   \eappto{@gls@nohyperlist}{, #1}%
748 }%
749 }%
750 }

```

`ignoredglossary` Starred form.

```

751 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
752   \ifcsdef{glolist@#1}
753   {%
754   {%
755     \ifdefempty{@ignored@glossaries
756     {%
757       \edef{@ignored@glossaries{#1}%
758     }%
759     {%
760       \eappto{@ignored@glossaries}{, #1}%
761     }%
762     \csgdef{glolist@#1}{,}%
763     \ifcsundef{gls@#1@entryfmt}%
764     {%
765       \defglsentryfmt[#1]{\glsentryfmt}%
766     }%
767     {}%
768   }%
769 }

```

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

```

770 \newcommand*{\glxtrcopytoglossary}[2]{%
771   \glsoifexists{#1}%
772   {%
773     \ifcsdef{glolist@#2}
774     {%
775       \cseappto{glolist@#2}{#1,%
776     }%
777     {%
778       \glxtrundefaction{Glossary type ‘#2’ doesn’t exist}{}%

```

```

779     }%
780 }%
781 }

```

1.3.1 Existence Checks

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

```

782 \renewcommand{\glsdoifexists}[2]{%
783   \ifglstryexists{#1}{#2}%
784   {%

```

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

```

785     \edef\glslabel{\glsdetoklabel{#1}}%
786     \glstrundefaction{Glossary entry '\glslabel'
787       has not been defined}{You need to define a glossary entry before
788       you can reference it.}%
789   }%
790 }

```

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

```

791 \renewcommand{\glsdoifnoexists}[2]{%
792   \ifglstryexists{#1}{%
793     \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
794       has already been defined}{}}{#2}%
795 }

```

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

```

796 \ifdef\glsdoifexistsordo
797 {%
798   \renewcommand{\glsdoifexistsordo}[3]{%
799     \ifglstryexists{#1}{#2}%
800     {%
801       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
802         has not been defined}{You need to define a glossary entry
803         before you can use it.}%
804       #3%
805     }%
806   }%
807 }
808 {%
809   \glstr@warnonexistsordo\glsdoifexistsordo
810   \newcommand{\glsdoifexistsordo}[3]{%
811     \ifglstryexists{#1}{#2}%
812     {%
813       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
814         has not been defined}{You need to define a glossary entry
815         before you can use it.}%

```

```

816      #3%
817    }%
818  }%
819 }

```

arynoexistsordo Similarly for \doifglossarynoexistsordo.

```

820 \ifdef\doifglossarynoexistsordo
821 {%
822   \renewcommand{\doifglossarynoexistsordo}[3]{%
823     \ifglossaryexists{#1}%
824     {%
825       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
826       #3%
827     }%
828     {#2}%
829   }%
830 }
831 {%
832   \glstr@warnonexistsordo\doifglossarynoexistsordo
833   \newcommand{\doifglossarynoexistsordo}[3]{%
834     \ifglossaryexists{#1}%
835     {%
836       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
837       #3%
838     }%
839     {#2}%
840   }%
841 }
842

```

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

```

843 \appto\@newglossaryentryposthook{%
844   \ifdefvoid\@glo@see
845     {\csxdef{glo@\@glo@label @see}{}}%
846     {%
847       \csxdef{glo@\@glo@label @see}{\@glo@see}%
848       \@glstr@autoindexcrossrefs
849     }%
850 }
851 \appto\@gls@keymap{,{see}{see}}

```

\glstrusesee Apply \glsseeformat to the see key if not empty.

```

852 \newcommand*{\glstrusesee}[1]{%
853   \glsdoifexists{#1}%
854   {%
855     \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
856     \ifdefempty\@glo@see
857       {}%
858     }%

```

```

859     \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
860   }%
861 }%
862 }

\glxtr@usesee
863 \newcommand*{\glxtr@usesee}[1][\seename]{%
864   \@glxtr@usesee[#1]%
865 }

\@glxtr@usesee
866 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
867   \glxtruseseeformat{#1}{#2}%
868 }

xtruseseeformat  The format used by \glxtrusesee. The first argument is the tag (such as \seename). The
                  second argument is the comma-separated list of cross-referenced labels.
869 \newcommand*{\glxtruseseeformat}[2]{%
870   \glssformat{#1}{#2}{}%
871 }

      Add all unused cross-references at the end of the document.
872 \AtEndDocument{\if@glxtrindexcrossrefs\glxtraddallcrossrefs\fi}

addallcrossrefs  Iterate through all used entries and if they have a cross-reference, make sure the cross-
                  reference has been added.
873 \newcommand*{\glxtraddallcrossrefs}{%
874   \forallglossaries{\@glo@type}%
875   {%
876     \forallgsentries[\@glo@type]{\@glo@label}%
877     {%
878       \ifglused{\@glo@label}{\@glxtr@addunusedxrefs{\@glo@label}}{}}%
879     }%
880   }%
881 }

@addunusedxrefs  If the given entry has a see field add all unused cross-references.
882 \newcommand*{\@glxtr@addunusedxrefs}[1]{%
883   \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
884   \ifdefvoid\@glo@see
885   {%
886     {%
887       \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
888     }%
889   }

lsxtr@addunused  Adds all the entries if they haven't been used.
890 \newcommand*{\glxtr@addunused}[1][{}]{%

```

```

891 \glxstr@addunused
892 }

```

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

```

893 \def\glxstr@addunused#1\endglxstr@addunused{%
894 \for\glxstr@label:=#1\do
895 {%
896 \ifglxsused{\glxstr@label}{}%
897 {%
898 \glsadd[format=glxstrunusedformat]{\glxstr@label}%
899 \glunset{\glxstr@label}%
900 \glxstr@addunusedxrefs{\glxstr@label}%
901 }%
902 }%
903 }

```

`glxstrunusedformat`

```

904 \newcommand*{\glxstrunusedformat}[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.

```

905 \let\glxstr@orgmakenoidxglossaries\makenoidxglossaries
906 \renewcommand{\makenoidxglossaries}{%
907 \glxstr@orgmakenoidxglossaries
908 \ifglxstrdocdefrestricted

```

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

```

909 \renewcommand*\gls@reference}[3]{%
910 \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
911 \ifinlistcs{##2}{\glsref@##1}%
912 {}%
913 {\listcsgadd{\glsref@##1}{##2}}%
914 \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
915 {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
916 {}%
917 \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
918 }%
919 \else

```

Disable document definitions.

```

920 \glxstrdocdeffalse
921 \fi
922 \disable@keys{glossaries-extra.sty}{docdef}%
923 }

```

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

```

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

     Restricted mode just needs to allow the see value.
938     \let\gls@checkseeallowed\relax
939   \fi
940 }%

```

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`

```

941 \newcommand*{\GlsXtrEnableOnTheFly}{%
942   \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
943 }

```

`\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.)

```

944 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
945   \renewcommand*{\glsdetoklabel}[1]{%
946     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
947     {%
948       \expandafter\detokenize\expandafter{##1}%
949     }%
950     {\detokenize{##1}}%
951   }%
952   \@GlsXtrEnableOnTheFly
953 }
954 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%

```

```

955 \expandafter\if\glsbackslash#1%
956   #3%
957 \else
958   #4%
959 \fi
960 }

```

sxtrstarflywarn

```

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

```

rEnableOnTheFly

```

966 \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

```

967 \newcommand*{\glsxtrcat}{general}

```

\glsxtr

```

968 \newcommand*{\glsxtr}[1] [] {%
969   \def\glsxtr@keylist{##1}%
970   \@glsxtr
971 }

```

\@glsxtr

```

972 \newcommand*{\@glsxtr}[2] [] {%
973   \ifglsentryexists{##2}%
974   {%
975     \ifblank{##1}{ }\{\GlsXtrWarning{##1}{##2}}%
976   }%
977   {%
978     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
979     description={\nopostdesc},##1}%
980   }%
981   \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
982 }

```

\Glsxtr

```

983 \newcommand*{\Glsxtr}[1] [] {%
984   \def\glsxtr@keylist{##1}%
985   \@Glsxtr
986 }

```



```

\@Glsxtr
987 \newcommand*{\@Glsxtr}[2] [] {%
988 \ifglentryexists{##2}%
989 {%
990 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
991 }%
992 {%
993 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
994 description={\nopostdesc},##1}%
995 }%
996 \expandafter\Gls\expandafter[\glstr@keylist]{##2}%
997 }

\glstrpl
998 \newcommand*{\glstrpl}[1] [] {%
999 \def\glstr@keylist{##1}%
1000 \@glstrpl
1001 }

\@glstrpl
1002 \newcommand*{\@glstrpl}[2] [] {%
1003 \ifglentryexists{##2}%
1004 {%
1005 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
1006 }%
1007 {%
1008 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1009 description={\nopostdesc},##1}%
1010 }%
1011 \expandafter\glsp\expandafter[\glstr@keylist]{##2}%
1012 }

\Glsxtrpl
1013 \newcommand*{\Glsxtrpl}[1] [] {%
1014 \def\glstr@keylist{##1}%
1015 \@Glsxtrpl
1016 }

\@Glsxtrpl
1017 \newcommand*{\@Glsxtrpl}[2] [] {%
1018 \ifglentryexists{##2}
1019 {%
1020 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
1021 }%
1022 {%
1023 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1024 description={\nopostdesc},##1}%
1025 }%
1026 \expandafter\Glspl\expandafter[\glstr@keylist]{##2}%

```

```
1027 }
```

`\GlsXtrWarning`

```
1028 \newcommand*{\GlsXtrWarning}[2]{%
1029   \def\@glxtr@optlist{##1}%
1030   \@onelevel@sanitize\@glxtr@optlist
1031   \GlossariesExtraWarning{The options ‘\@glxtr@optlist’ have
1032     been ignored for entry ‘##2’ as it has already been defined}%
1033 }
```

Disable commands after the glossary:

```
1034 \renewcommand\@printglossary[2]{%
1035   \def\@glxtr@printglossopts{##1}%
1036   \@glxtr@orgprintglossary{##1}{##2}%
1037   \def\@glxtr{\@glxtr@disabledflycommand\glxtr}%
1038   \def\@glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
1039   \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
1040   \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
1041 }
```

`abledflycommand`

```
1042 \newcommand*{\@glxtr@disabledflycommand}[1]{%
1043   \PackageError{glossaries-extra}%
1044   {\string##1\space can’t be used after any of the \MessageBreak
1045     glossaries have been displayed}%
1046   {The on-the-fly commands enabled by
1047     \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1048     before the glossaries. If you want to use any entries \MessageBreak
1049     after any of the glossaries, you must use the standard \MessageBreak
1050     method of first defining the entry and then using the \MessageBreak
1051     entry with commands like \string\glx}%
1052   \@glxtr@disabledflycommand
1053 }%
1054 \newcommand*{\@glxtr@disabledflycommand}[2][\@glxtr@disabledflycommand]{##2}
```

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

```
1055 \let\GlsXtrEnableOnTheFly\relax
1056 }
1057 \@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.

```
1058 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}
```

Modify `\setglossarystyle` to set the above.

etglossarystyle

```
1059 \renewcommand*{\setglossarystyle}[1]{%
1060   \ifcsundef{@glsstyle@#1}%
1061   {%
1062     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
1063   }%
1064   {%
1065     \csname @glsstyle@#1\endcsname
```

Only set the current style if it exists.

```
1066     \protected@edef\@glsxtr@current@style{#1}%
1067   }%
1068   \ifx\@glossary@default@style\relax
1069     \protected@edef\@glossary@default@style{#1}%
1070   \fi
1071 }
```

In case we have an old version of glossaries:

```
1072 \ifdef\@glossary@default@style
1073 {}
1074 {%
1075   \let\@glossary@default@style\relax
1076 }
```

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

```
1077 \ifdef\glslistdottedwidth
1078 {%
1079   \ifdim\glslistdottedwidth=.5\hsize
1080     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1081     \AtBeginDocument{%
1082       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1083         \setlength{\glslistdottedwidth}{.5\columnwidth}%
1084       \fi
1085     }%
1086   \fi
1087 }
1088 {}%
```

Similarly for `\glsdescwidth`:

`\glsdescwidth`

```
1089 \ifdef\glsdescwidth
1090 {%
1091   \ifdim\glsdescwidth=.6\hsize
1092     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1093     \AtBeginDocument{%
1094       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1095         \setlength{\glsdescwidth}{.6\columnwidth}%
1096     }%
1097   \fi
1098 }
```

```

1096     \fi
1097   }%
1098   \fi
1099 }
1100 {}%

```

and for \glspagelistwidth:

lspagelistwidth

```

1101 \ifdef\glspagelistwidth
1102 {%
1103   \ifdim\glspagelistwidth=.1\hsize
1104     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
1105     \AtBeginDocument{%
1106       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1107         \setlength{\glspagelistwidth}{.1\columnwidth}%
1108       \fi
1109     }%
1110   \fi
1111 }
1112 {}%

```

aryentrynumbers Has the nonnumberlist option been used?

```

1113 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1114 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1115   \glsnonnumberlistfalse
1116   \renewcommand*{\glossaryentrynumbers}[1]{%
1117     \ifglentryexists{\glscurrententrylabel}%
1118     {%
1119       \@glxtrpreloctag
1120       \GlsXtrFormatLocationList{#1}%
1121       \@glxtrpostloctag
1122       \gls@save@numberlist{#1}%
1123     }{}%
1124   }%
1125 \else
1126   \glsnonnumberlisttrue
1127   \renewcommand*{\glossaryentrynumbers}[1]{%
1128     \ifglentryexists{\glscurrententrylabel}%
1129     {%
1130       \gls@save@numberlist{#1}%
1131     }{}%
1132   }%
1133 \fi

```

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

```

1134 \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`

```

1135 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
1136   \let\@glsxtrpreloctag\@glsxtrpreloctag
1137   \let\@glsxtrpostloctag\@glsxtrpostloctag
1138   \renewcommand*{\@glsxtr@pagetag}{#1}%
1139   \renewcommand*{\@glsxtr@pagetag}{#2}%
1140   \renewcommand*{\@glsxtr@savepreloctag}[2]{%
1141     \csgdef{\@glsxtr@preloctag##1}{##2}%
1142   }%
1143   \renewcommand*{\@glsxtr@doloctag}{%
1144     \ifcsundef{\@glsxtr@preloctag\@glscurrententrylabel}%
1145     {%
1146       \GlossariesWarning{Missing pre-location tag for ‘\@glscurrententrylabel’.
1147         Rerun required}%
1148     }%
1149     {%
1150       \csuse{\@glsxtr@preloctag\@glscurrententrylabel}%
1151     }%
1152   }%
1153 }
1154 \@onlypreamble\GlsXtrEnablePreLocationTag

```

`glsxtrpreloctag`

```

1155 \newcommand*{\@glsxtrpreloctag}{%
1156   \let\@glsxtr@org@delimN\delimN
1157   \let\@glsxtr@org@delimR\delimR
1158   \let\@glsxtr@org@glsgignore\glsgignore
1159   \gdef is required as the delimiters may occur inside a scope.
1159   \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1160   \renewcommand*{\delimN}{%
1161     \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1162     \@glsxtr@org@delimN}%
1163   \renewcommand*{\delimR}{%
1164     \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1165     \@glsxtr@org@delimR}%
1166   \renewcommand*{\glsgignore}[1]{%
1167     \gdef\@glsxtr@thisloctag{\relax}%
1168     \@glsxtr@org@glsgignore{##1}}%
1169   \@glsxtr@doloctag
1170 }

```

`glsxtrpreloctag`

```

1171 \newcommand*{\@glsxtrpreloctag}{}

```

@glsxtr@pagetag

```
1172 \newcommand*{\@glsxtr@pagetag}{}%
```

glsxtr@pagetag

```
1173 \newcommand*{\@glsxtr@pagetag}{}%
```

lsxtrpostloctag

```
1174 \newcommand*{\@glsxtrpostloctag}{%
1175   \let\delimN\@glsxtr@org@delimN
1176   \let\delimR\@glsxtr@org@delimR
1177   \let\glsignore\@glsxtr@org@glsignore
1178   \protected@write\@auxout{}{%
1179     {\string\@glsxtr@savepreloctag{\glscurrententrylabel}\@glsxtr@thisloctag}}%
1180 }
```

lsxtrpostloctag

```
1181 \newcommand*{\@glsxtrpostloctag}{}
```

lsxtr@preloctag

```
1182 \newcommand*{\@glsxtr@savepreloctag}[2]{%
1183 \protected@write\@auxout{}{%
1184   \string\providecommand\string\@glsxtr@savepreloctag[2]{}}
```

glsxtr@doloctag

```
1185 \newcommand*{\@glsxtr@doloctag}{}
```

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

```
1186 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
1187   \XKV@plfalse
1188   \XKV@sttrue
1189   \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
1190   {%
1191     \csname glsnonumberlist\XKV@resa\endcsname
1192     \ifglsnonumberlist
1193       \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1194     \else
1195       \def\glossaryentrynumbers##1{%
1196         \@glsxtrpreloctag
1197         \GlsXtrFormatLocationList{##1}%
1198         \@glsxtrpostloctag
1199         \gls@save@numberlist{##1}}%
1200     \fi
1201   }%
1202 }
```

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.

```
1203 \renewcommand*{\glsentryfmt}{%
1204   \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}}{}%
1205   \glsifregular{\glslabel}%
1206   {\glstrregularfont{\glsgenentryfmt}}%
1207   {%
1208     \ifglshasshort{\glslabel}%
1209     {\glstrgenabbrvfmt}%
1210     {\glstrregularfont{\glsgenentryfmt}}%
1211   }%
1212 }
```

`glstrregularfont` Font used for regular entries.

```
1213 \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.

```
1214 \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.

```
1215 \@glstr@record{#2}{#3}%
1216 \glstoifexists{#3}%
1217 {%
```

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).

```
1218   \let\glstrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
1219   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1220   \def\glscustomtext{#4}%
1221   \@glstr@field@linkdefs
1222   #1%
1223   \@gls@link[#2]{#3}{#4}%
1224   \let\ifKV@glslink@hyper\glstrorg@ifKV@glslink@hyper
1225 }%
```

```

1226 \glspostlinkhook
1227 }

```

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

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

```

1228 \let\@glsxtr@org@gls@\@gls@
1229 \def\@gls@#1#2{%
1230   \@glsxtr@record{#1}{#2}%
1231   \@glsxtr@org@gls@{#1}{#2}%
1232 }%

```

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

```

1233 \let\@glsxtr@org@glspl@\@glspl@
1234 \def\@glspl@#1#2{%
1235   \@glsxtr@record{#1}{#2}%
1236   \@glsxtr@org@glspl@{#1}{#2}%
1237 }%

```

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

```

1238 \let\@glsxtr@org@Gls@\@Gls@
1239 \def\@Gls@#1#2{%
1240   \@glsxtr@record{#1}{#2}%
1241   \@glsxtr@org@Gls@{#1}{#2}%
1242 }%

```

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

```

1243 \let\@glsxtr@org@Glspl@\@Glspl@
1244 \def\@Glspl@#1#2{%
1245   \@glsxtr@record{#1}{#2}%
1246   \@glsxtr@org@Glspl@{#1}{#2}%
1247 }%

```

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

```

1248 \let\@glsxtr@org@GLS@\@GLS@
1249 \def\@GLS@#1#2{%
1250   \@glsxtr@record{#1}{#2}%
1251   \@glsxtr@org@GLS@{#1}{#2}%
1252 }%

```

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

```

1253 \let\@glsxtr@org@GLSpl@\@GLSpl@
1254 \def\@GLSpl@#1#2{%
1255   \@glsxtr@record{#1}{#2}%
1256   \@glsxtr@org@GLSpl@{#1}{#2}%
1257 }%

```


\@glsdispl Save the original definition and redefine.

```
1258 \let\@glsxtr@org@glsdispl\@glsdispl
1259 \renewcommand*{\@glsdispl}[3][]{%
1260   \@glsxtr@record{#1}{#2}%
1261   \@glsxtr@org@glsdispl[#1]{#2}{#3}%
1262 }
```

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

```
1263 \renewcommand*{\@gls@link}[3][]{%
1264   \@glsxtr@record{#1}{#2}%
1265   \glsdoifexistsord{#2}%
1266   {%
1267     \let\do@gls@link@checkfirsthyper\relax
1268     \@gls@link[#1]{#2}{#3}%
1269   }%
1270   {%
1271     \glstextformat{#3}%
1272   }%
1273   \glspostlinkhook
1274 }
```

\glsadd Redefine to include \@glsxtr@record

```
1275 \renewrobustcmd*{\glsadd}[2][]{%
1276   \@gls@adjustmode
1277   \@glsxtr@record{#1}{#2}%
1278   \glsdoifexists{#2}%
1279   {%
1280     \def\@glsnumberformat{glsnumberformat}%
1281     \edef\@gls@counter{\csname glo@\glsdetoklabel{#2}@counter\endcsname}%
1282     \setkeys{glossadd}{#1}%
1283     \@gls@saveentrycounter
1284     \@do@wrglossary{#2}%
1285   }%
1286 }
```

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

```
1287 \newcommand*{\@glsxtr@field@linkdefs}{%
1288   \let\glsxtrifwasfirstuse\@secondoftwo
1289   \let\glsifplural\@secondoftwo
1290   \let\glsifcaps\@firstofthree
1291   \let\glsinsert\@empty
1292 }
```

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

assignfieldfont

```
1293 \newcommand*{\glsxtrassignfieldfont}[1]{%
1294   \ifglstryexists{#1}%
```

```

1295 {%
1296   \ifglshasshort{#1}%
1297   {%
1298     \glsetabbrvfmt{\glscategory{#1}}%
1299     \glusifregular{#1}%
1300     {\let\@gls@field@font\glxtrregularfont}%
1301     {\let\@gls@field@font\@firstofone}%
1302   }%
1303   {%
1304     \glusifnotregular{#1}%
1305     {\let\@gls@field@font\@firstofone}%
1306     {\let\@gls@field@font\glxtrregularfont}%
1307   }%
1308 }%
1309 {%
1310   \let\@gls@field@font\@gobble
1311 }%
1312 }

```

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

```

1313 \def\@glstext@#1#2[#3]{%
1314   \glxtrassignfieldfont{#2}%
1315   \@gls@field@link{#1}{#2}{\@gls@field@font{\glssaccesstext{#2}#3}}%
1316 }

```

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

```

1317 \def\@GLStext@#1#2[#3]{%
1318   \glxtrassignfieldfont{#2}%
1319   \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
1320   {\@gls@field@font{\@GLSsaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
1321 }

```

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

```

1322 \def\@Glstext@#1#2[#3]{%
1323   \glxtrassignfieldfont{#2}%
1324   \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
1325   {\@gls@field@font{\@Glsaccesstext{#2}#3}}%
1326 }

```

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

`\ecknohyperfirst`

```

1327 \newcommand*\glxtrchecknohyperfirst}[1]{%
1328   \glusifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
1329 }

```

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

```

1330 \def\@glsfirst@#1#2[#3]{%
1331   \glxtrassignfieldfont{#2}%

```

Ensure that \glsfirst honours the nohyperfirst attribute.

```

1332 \@gls@field@link
1333 [\let\glsxtrifwasfirstuse\@firstoftwo
1334 \glsxtrchecknohyperfirst{#2}%
1335 ]{#1}{#2}%
1336 {\@gls@field@font{\glsaccessfirst{#2}#3}}%
1337 }

```

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

```

1338 \def\@Glsfirst@#1#2[#3]{%
1339 \glsxtrassignfieldfont{#2}%
1340 \@gls@field@link
1341 [\let\glsxtrifwasfirstuse\@firstoftwo
1342 \let\glsacapscase\@secondofthree
1343 \glsxtrchecknohyperfirst{#2}%
1344 ]%
1345 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
1346 }

```

Ensure that \Glsfirst honours the nohyperfirst attribute.

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

```

1347 \def\@GLSfirst@#1#2[#3]{%
1348 \glsxtrassignfieldfont{#2}%
1349 \@gls@field@link
1350 [\let\glsxtrifwasfirstuse\@firstoftwo
1351 \let\glsacapscase\@thirdofthree
1352 \glsxtrchecknohyperfirst{#2}%
1353 ]%
1354 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1355 }

```

Ensure that \GLSfirst honours the nohyperfirst attribute.

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

```

1356 \def\@glsplural@#1#2[#3]{%
1357 \glsxtrassignfieldfont{#2}%
1358 \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1359 {\@gls@field@font{\glsaccessplural{#2}#3}}%
1360 }

```

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

```

1361 \def\@Glsplural@#1#2[#3]{%
1362 \glsxtrassignfieldfont{#2}%
1363 \@gls@field@link
1364 [\let\glsifplural\@firstoftwo
1365 \let\glsacapscase\@secondofthree
1366 ]%
1367 {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1368 }

```

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

```
1369 \def\@GLSplural@#1#2[#3]{%
1370   \glstrassignfieldfont{#2}%
1371   \@gls@field@link
1372   [\let\glsifplural\@firstoftwo
1373   \let\glscapscase\@thirdofthree
1374   ]%
1375   {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1376 }
```

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

```
1377 \def\@glsfirstplural@#1#2[#3]{%
1378   \glstrassignfieldfont{#2}%
1379   \@gls@field@link
1380   [\let\glstrifwasfirstuse\@firstoftwo
1381   \let\glsifplural\@firstoftwo
1382   \glstrchecknohyperfirst{#2}%
1383   ]%
1384   {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1385 }
```

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

```
1386 \def\@GLsfirstplural@#1#2[#3]{%
1387   \glstrassignfieldfont{#2}%
1388   \@gls@field@link
1389   [\let\glstrifwasfirstuse\@firstoftwo
1390   \let\glsifplural\@firstoftwo
1391   \let\glscapscase\@secondofthree
1392   \glstrchecknohyperfirst{#2}%
1393   ]%
1394   {#1}{#2}{\@gls@field@font{\GLsaccessfirstplural{#2}#3}}%
1395 }
```

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

```
1396 \def\@GLSfirstplural@#1#2[#3]{%
1397   \glstrassignfieldfont{#2}%
1398   \@gls@field@link
1399   [\let\glstrifwasfirstuse\@firstoftwo
1400   \let\glsifplural\@firstoftwo
1401   \let\glscapscase\@thirdofthree
1402   \glstrchecknohyperfirst{#2}%
1403   ]%
1404   {#1}{#2}%
1405   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
1406 }
```

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

```
1407 \def\@glsname@#1#2[#3]{%
1408   \glstrassignfieldfont{#2}%
1409   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
1410 }
```

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

```
1411 \def\@Glsname@#1#2[#3]{%
1412   \glstrassignfieldfont{#2}%
1413   \@gls@field@link
1414   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1415   {\@gls@field@font{\Glsaccessname{#2}#3}}%
1416 }
```

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

```
1417 \def\@GLSname@#1#2[#3]{%
1418   \glstrassignfieldfont{#2}%
1419   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1420   {#1}{#2}%
1421   {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
1422 }
```

\@glsdesc@

```
1423 \def\@glsdesc@#1#2[#3]{%
1424   \glstrassignfieldfont{#2}%
1425   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
1426 }
```

\@Glsdesc@ First letter uppercase version.

```
1427 \def\@Glsdesc@#1#2[#3]{%
1428   \glstrassignfieldfont{#2}%
1429   \@gls@field@link
1430   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1431   {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
1432 }
```

\@GLSdesc@ All uppercase version.

```
1433 \def\@GLSdesc@#1#2[#3]{%
1434   \glstrassignfieldfont{#2}%
1435   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1436   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
1437 }
```

@glsdescplural@ No case-changing version.

```
1438 \def\@glsdescplural@#1#2[#3]{%
1439   \glstrassignfieldfont{#2}%
1440   \@gls@field@link
1441   [\let\glscapscase\@secondoftwo
```

```

1442 \let\glsifplural\@firstoftwo
1443 ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
1444 }

```

@Glsdescplural@ First letter uppercase version.

```

1445 \def\@Glsdescplural@#1#2[#3]{%
1446 \glsxtrassignfieldfont{#2}%
1447 \@gls@field@link
1448 [\let\glscapscase\@secondoftwo
1449 \let\glsifplural\@firstoftwo
1450 ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
1451 }

```

@GLSdescplural@ All uppercase version.

```

1452 \def\@GLSdesc@#1#2[#3]{%
1453 \glsxtrassignfieldfont{#2}%
1454 \@gls@field@link
1455 [\let\glscapscase\@thirdoftwo
1456 \let\glsifplural\@firstoftwo
1457 ]%
1458 {#1}{#2}%
1459 {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
1460 }

```

\@glssymbol@

```

1461 \def\@glssymbol@#1#2[#3]{%
1462 \glsxtrassignfieldfont{#2}%
1463 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
1464 }

```

\@Glsymbol@ First letter uppercase version.

```

1465 \def\@Glsymbol@#1#2[#3]{%
1466 \glsxtrassignfieldfont{#2}%
1467 \@gls@field@link
1468 [\let\glscapscase\@secondoftwo]%
1469 {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
1470 }

```

\@GLSsymbol@ All uppercase version.

```

1471 \def\@GLSsymbol@#1#2[#3]{%
1472 \glsxtrassignfieldfont{#2}%
1473 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1474 {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
1475 }

```

lssymbolplural@ No case-changing version.

```

1476 \def\@glssymbolplural@#1#2[#3]{%
1477 \glsxtrassignfieldfont{#2}%

```

```

1478 \@gls@field@link
1479 [\let\glscapscase\@secondoftwo
1480 \let\glsifplural\@firstoftwo
1481 ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}{#3}}}%
1482 }

```

Lssymbolplural@ First letter uppercase version.

```

1483 \def\@GLssymbolplural@#1#2[#3]{%
1484 \glstrassignfieldfont{#2}%
1485 \@gls@field@link
1486 [\let\glscapscase\@secondoftwo
1487 \let\glsifplural\@firstoftwo
1488 ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}{#3}}}%
1489 }

```

LSsymbolplural@ All uppercase version.

```

1490 \def\@GLSsymbol@#1#2[#3]{%
1491 \glstrassignfieldfont{#2}%
1492 \@gls@field@link
1493 [\let\glscapscase\@thirdoftwo
1494 \let\glsifplural\@firstoftwo
1495 ]%
1496 {#1}{#2}%
1497 {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
1498 }

```

\@Glsuseri@ First letter uppercase version.

```

1499 \def\@Glsuseri@#1#2[#3]{%
1500 \glstrassignfieldfont{#2}%
1501 \@gls@field@link
1502 [\let\glscapscase\@secondoftwo]{#1}{#2}%
1503 {\@gls@field@font{\Glsentryuseri{#2}{#3}}}%
1504 }

```

\@GLSuseri@ All uppercase version.

```

1505 \def\@GLSuseri@#1#2[#3]{%
1506 \glstrassignfieldfont{#2}%
1507 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1508 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glsentryuseri{#2}{#3}}}%
1509 }

```

\@Glsuserii@ First letter uppercase version.

```

1510 \def\@Glsuserii@#1#2[#3]{%
1511 \glstrassignfieldfont{#2}%
1512 \@gls@field@link
1513 [\let\glscapscase\@secondoftwo]%
1514 {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}{#3}}}%
1515 }

```

\@GLSuserii@ All uppercase version.

```
1516 \def\@GLSuserii@#1#2[#3]{%
1517   \glstrassignfieldfont{#2}%
1518   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1519   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
1520 }
```

\@Glsuseriii@ First letter uppercase version.

```
1521 \def\@Glsuseriii@#1#2[#3]{%
1522   \glstrassignfieldfont{#2}%
1523   \@gls@field@link
1524   [\let\glscapscase\@secondoftwo]%
1525   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
1526 }
```

\@GLSuseriii@ All uppercase version.

```
1527 \def\@GLSuseriii@#1#2[#3]{%
1528   \glstrassignfieldfont{#2}%
1529   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1530   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
1531 }
```

\@Glsuseriv@ First letter uppercase version.

```
1532 \def\@Glsuseriv@#1#2[#3]{%
1533   \glstrassignfieldfont{#2}%
1534   \@gls@field@link
1535   [\let\glscapscase\@secondoftwo]%
1536   {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
1537 }
```

\@GLSuseriv@ All uppercase version.

```
1538 \def\@GLSuseriv@#1#2[#3]{%
1539   \glstrassignfieldfont{#2}%
1540   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1541   {#1}{#2}%
1542   {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
1543 }
```

\@Glsuserv@ First letter uppercase version.

```
1544 \def\@Glsuserv@#1#2[#3]{%
1545   \glstrassignfieldfont{#2}%
1546   \@gls@field@link
1547   [\let\glscapscase\@secondoftwo]%
1548   {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
1549 }
```

\@GLSuserv@ All uppercase version.

```
1550 \def\@GLSuserv@#1#2[#3]{%
```



```

1551 \glstrassignfieldfont{#2}%
1552 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1553 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
1554 }

```

\@Glsuservi@ First letter uppercase version.

```

1555 \def\@Glsuservi@#1#2[#3]{%
1556 \glstrassignfieldfont{#2}%
1557 \@gls@field@link
1558 [\let\glscapscase\@secondoftwo]%
1559 {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}}%
1560 }

```

\@GLSuservi@ All uppercase version.

```

1561 \def\@GLSuservi@#1#2[#3]{%
1562 \glstrassignfieldfont{#2}%
1563 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1564 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
1565 }

```

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

\@acrshort No case change.

```

1566 \def\@acrshort#1#2[#3]{%
1567 \glsdoifexists{#2}%
1568 {%
1569 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1570 \let\glxtrifwasfirstuse\@secondoftwo
1571 \let\glsifplural\@secondoftwo
1572 \let\glscapscase\@firstofthree
1573 \let\glsinsert\@empty
1574 \def\glscustomtext{%
1575 \acronymfont{\glsaccesssshort{#2}}#3%
1576 }%
1577 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1578 }%
1579 \glspostlinkhook
1580 }

```

\@Acrshort First letter uppercase.

```

1581 \def\@Acrshort#1#2[#3]{%
1582 \glsdoifexists{#2}%
1583 {%
1584 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1585 \let\glxtrifwasfirstuse\@secondoftwo
1586 \let\glsifplural\@secondoftwo
1587 \let\glscapscase\@secondofthree
1588 \let\glsinsert\@empty

```

```

1589 \def\glscustomtext{%
1590 \acronymfont{\Glsaccesssshort{#2}}#3%
1591 }%
1592 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1593 }%
1594 \glspostlinkhook
1595 }

```

\@ACRshort All uppercase.

```

1596 \def\@ACRshort#1#2[#3]{%
1597 \glsdofexists{#2}%
1598 {%
1599 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1600 \let\glstrifwasfirstuse\@secondoftwo
1601 \let\glsifplural\@secondoftwo
1602 \let\glscapscase\@thirdofthree
1603 \let\glsinsert\@empty
1604 \def\glscustomtext{%
1605 \mfirstucMakeUppercase{\acronymfont{\Glsaccesssshort{#2}}#3}%
1606 }%
1607 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1608 }%
1609 \glspostlinkhook
1610 }

```

\@acrshortpl No case change.

```

1611 \def\@acrshortpl#1#2[#3]{%
1612 \glsdofexists{#2}%
1613 {%
1614 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1615 \let\glstrifwasfirstuse\@secondoftwo
1616 \let\glsifplural\@firstoftwo
1617 \let\glscapscase\@firstofthree
1618 \let\glsinsert\@empty
1619 \def\glscustomtext{%
1620 \acronymfont{\Glsaccesssshortpl{#2}}#3%
1621 }%
1622 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1623 }%
1624 \glspostlinkhook
1625 }

```

\@Acrshortpl First letter uppercase.

```

1626 \def\@Acrshortpl#1#2[#3]{%
1627 \glsdofexists{#2}%
1628 {%
1629 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1630 \let\glstrifwasfirstuse\@secondoftwo
1631 \let\glsifplural\@firstoftwo

```

```

1632 \let\glscapscase\@secondofthree
1633 \let\glsinsert\@empty
1634 \def\glscustomtext{%
1635     \acronymfont{\Glsaccesssshortpl{#2}}#3%
1636 }%
1637 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1638 }%
1639 \glspostlinkhook
1640 }

```

\@ACRshortpl All uppercase.

```

1641 \def\@ACRshortpl#1#2[#3]{%
1642     \glsdoifexists{#2}%
1643     {%
1644         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1645         \let\glstrifwasfirstuse\@secondoftwo
1646         \let\glsifplural\@firstoftwo
1647         \let\glscapscase\@thirdofthree
1648         \let\glsinsert\@empty
1649         \def\glscustomtext{%
1650             \mfirstucMakeUppercase{\acronymfont{\Glsaccesssshortpl{#2}}#3}%
1651         }%
1652         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1653     }%
1654     \glspostlinkhook
1655 }

```

\@acrlong No case change.

```

1656 \def\@acrlong#1#2[#3]{%
1657     \glsdoifexists{#2}%
1658     {%
1659         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1660         \let\glstrifwasfirstuse\@secondoftwo
1661         \let\glsifplural\@secondoftwo
1662         \let\glscapscase\@firstofthree
1663         \let\glsinsert\@empty
1664         \def\glscustomtext{%
1665             \acronymfont{\Glsaccesslong{#2}}#3%
1666         }%
1667         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1668     }%
1669     \glspostlinkhook
1670 }

```

\@Acrlong First letter uppercase.

```

1671 \def\@Acrlong#1#2[#3]{%
1672     \glsdoifexists{#2}%
1673     {%
1674         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```

```

1675 \let\glxtrifwasfirstuse\@secondoftwo
1676 \let\glsifplural\@secondoftwo
1677 \let\glscapscase\@secondofthree
1678 \let\glsinsert\@empty
1679 \def\glscustomtext{%
1680 \acronymfont{\Glsaccesslong{#2}}#3%
1681 }%
1682 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1683 }%
1684 \glspostlinkhook
1685 }

```

\@ACRlong All uppercase.

```

1686 \def\@ACRlong#1#2[#3]{%
1687 \glsdoifexists{#2}%
1688 {%
1689 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1690 \let\glxtrifwasfirstuse\@secondoftwo
1691 \let\glsifplural\@secondoftwo
1692 \let\glscapscase\@thirdofthree
1693 \let\glsinsert\@empty
1694 \def\glscustomtext{%
1695 \mfirstucMakeUppercase{\acronymfont{\glsaccesslong{#2}}#3}%
1696 }%
1697 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1698 }%
1699 \glspostlinkhook
1700 }

```

\@acrlongpl No case change.

```

1701 \def\@acrlongpl#1#2[#3]{%
1702 \glsdoifexists{#2}%
1703 {%
1704 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1705 \let\glxtrifwasfirstuse\@secondoftwo
1706 \let\glsifplural\@firstoftwo
1707 \let\glscapscase\@firstofthree
1708 \let\glsinsert\@empty
1709 \def\glscustomtext{%
1710 \acronymfont{\glsaccesslongpl{#2}}#3%
1711 }%
1712 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1713 }%
1714 \glspostlinkhook
1715 }

```

\@Acrlongpl First letter uppercase.

```

1716 \def\@Acrlongpl#1#2[#3]{%
1717 \glsdoifexists{#2}%

```

```

1718 {%
1719   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1720   \let\glxtrifwasfirstuse\@secondoftwo
1721   \let\gl@sifplural\@firstoftwo
1722   \let\glscapscase\@secondofthree
1723   \let\gl$insert\@empty
1724   \def\glscustomtext{%
1725     \acronymfont{\Glsaccesslongpl{#2}}#3%
1726   }%
1727   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1728 }%
1729 \glspostlinkhook
1730 }

```

\@ACRlongpl All uppercase.

```

1731 \def\@ACRlongpl#1#2[#3]{%
1732   \gl@dofexists{#2}%
1733   {%
1734     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1735     \let\glxtrifwasfirstuse\@secondoftwo
1736     \let\gl@sifplural\@firstoftwo
1737     \let\glscapscase\@thirdofthree
1738     \let\gl$insert\@empty
1739     \def\glscustomtext{%
1740       \mfirstucMakeUppercase{\acronymfont{\Glsaccesslongpl{#2}}#3}%
1741     }%
1742     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1743   }%
1744   \glspostlinkhook
1745 }

```

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

\@glsaddkey

```

1746 \renewcommand*{\@glsaddkey}[7]{%
1747   \key@ifundefined{glossentry}{#1}%
1748   {%
1749     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1750     \appto\@gl@keymap{,{#1}{#1}}%
1751     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
1752     \appto\@newglossaryentryposthook{%
1753       \letcs{\@glo@tmp}{@glo@#1}%
1754       \gl@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1755     }%
1756     \newcommand*{#3}[1]{\@gl@entry@field{##1}{#1}}%
1757     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

1758   \ifcsdef{@gls@user@#1@}%

```

```

1759 {%
1760   \PackageError{glossaries}%
1761   {Can't define '\string#5' as helper command
1762   '\expandafter\string\csname @gls@user@#1@\endcsname' already
1763   exists}%
1764   {}}%
1765 }%
1766 {%
1767   \expandafter\newcommand\expandafter*\expandafter
1768   {\csname @gls@user@#1@\endcsname}[2][{}]{%
1769     \new@ifnextchar[%
1770       {\csuse{@gls@user@#1@}{##1}{##2}}%
1771       {\csuse{@gls@user@#1@}{##1}{##2}[{}]}%
1772     \csdef{@gls@user@#1@}##1##2[##3]{%
1773       \@gls@field@link{##1}{##2}{#3{##2}##3}%
1774     }%
1775     \newrobustcmd*{#5}{%
1776       \expandafter\@gls@hyp@opt\csname @gls@user@#1@\endcsname}%
1777   }%

```

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

```

1778 \ifcsdef{@Gls@user@#1@}%
1779 {%
1780   \PackageError{glossaries}%
1781   {Can't define '\string#6' as helper command
1782   '\expandafter\string\csname @Gls@user@#1@\endcsname' already
1783   exists}%
1784   {}}%
1785 }%
1786 {%
1787   \expandafter\newcommand\expandafter*\expandafter
1788   {\csname @Gls@user@#1@\endcsname}[2][{}]{%
1789     \new@ifnextchar[%
1790       {\csuse{@Gls@user@#1@}{##1}{##2}}%
1791       {\csuse{@Gls@user@#1@}{##1}{##2}[{}]}%
1792     \csdef{@Gls@user@#1@}##1##2[##3]{%
1793       \@gls@field@link[\let\glscaps@case\@secondofthree]%
1794       {##1}{##2}{#4{##2}##3}%
1795     }%
1796     \newrobustcmd*{#6}{%
1797       \expandafter\@gls@hyp@opt\csname @Gls@user@#1@\endcsname}%
1798   }%

```

Finally the all caps version (modified):

```

1799 \ifcsdef{@GLS@user@#1@}%
1800 {%
1801   \PackageError{glossaries}%
1802   {Can't define '\string#7' as helper command
1803   '\expandafter\string\csname @GLS@user@#1@\endcsname' already
1804   exists}%

```

```

1805     {}%
1806 }%
1807 {%
1808     \expandafter\newcommand\expandafter*\expandafter
1809     {\csname @GLS@user@#1\endcsname}[2][{}]{%
1810         \new@ifnextchar[%
1811             {\csuse{@GLS@user@#1@}{##1}{##2}}}%
1812             {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%
1813     \csdef{@GLS@user@#1@}##1##2[##3]{%
1814         \@gls@field@link[\let\gls@scaps@case\@thirdofthree]%
1815         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}}%
1816     }%
1817     \newrobustcmd*{#7}{%
1818         \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
1819     }%
1820 }%
1821 {%
1822     \PackageError{glossaries-extra}{Key ‘#1’ already exists}{}%
1823 }%
1824 }

```

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

```

1825 \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.

```

1826 \let\@gls@xtr@org@checkfirsthyper\@gls@link@checkfirsthyper
1827 \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.

```

1828 \ifglsused{\glslabel}%
1829 {\let\gls@xtrifwasfirstuse\@secondoftwo}
1830 {\let\gls@xtrifwasfirstuse\@firstoftwo}%

```

Store the category label for convenience.

```

1831 \edef\gls@categorylabel{\gls@category{\glslabel}}%
1832 \ifglsused{\glslabel}%
1833 {%
1834     \glsifcategoryattribute{\gls@categorylabel}{nohypernext}{true}%
1835     {\KV@glslink@hyperfalse}}%
1836 }%
1837 {%
1838     \glsifcategoryattribute{\gls@categorylabel}{nohyperfirst}{true}%
1839     {\KV@glslink@hyperfalse}}%
1840 }%
1841 \glslinkcheckfirsthyperhook
1842 }

```

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.

```

1843 \ifdef\do@glsglssdisablehyperinlist
1844 {%
1845   \let\@glsglssxtr@do@glsglssdisablehyperinlist\do@glsglssdisablehyperinlist
1846   \renewcommand*{\do@glsglssdisablehyperinlist}{%
1847     \@glsglssxtr@do@glsglssdisablehyperinlist
1848     \glsglssifattribute{\glsglsslabel}{nohyper}{true}{\KV@glsglsslink@hyperfalse}}}%
1849   }
1850 }
1851 {}

```

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

```

1852 \define@boolkey{\glsglsslink}{noindex}[true]{}
1853 \KV@glsglsslink@noindexfalse

```

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

lt@glsglsslink@opts

```

1854 \ifdef\@glsglss@setdefault@glsglsslink@opts
1855 {
1856   \renewcommand*{\@glsglss@setdefault@glsglsslink@opts}{%
1857     \KV@glsglsslink@noindexfalse
1858     \@glsglssxtrsetaliasnoindex
1859   }
1860 }
1861 {

```

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

```

1862   \newcommand*{\@glsglss@setdefault@glsglsslink@opts}{%
1863     \KV@glsglsslink@noindexfalse
1864     \@glsglssxtrsetaliasnoindex
1865   }
1866   \pretoto\do@glsglssdisablehyperinlist{\@glsglss@setdefault@glsglsslink@opts}
1867 }

```

setaliasnoindex Allow user to hook into the alias noindex setting. Default behaviour switches off indexing for aliases.

```

1868 \newcommand*{\glsglssxtrsetaliasnoindex}{%
1869   \KV@glsglsslink@noindextrue
1870 }

```

setaliasnoindex

```

1871 \newcommand*{\@glsglssxtrsetaliasnoindex}{%
1872   \ifglsglsshasfield{alias}{\glsglsslabel}%
1873   {%
1874     \let\glsglssxtrindexaliased\@glsglssxtrindexaliased
1875     \glsglssxtrsetaliasnoindex

```



```

1876 \let\glxtrindexaliased\@no@glxtrindexaliased
1877 }%
1878 {}%
1879 }

```

xtrindexaliased

```

1880 \newcommand{\@glxtrindexaliased}{%
1881 \ifKV@glslink@noindex
1882 \else
1883 \begingroup
1884 \def\@glslnumberformat{glslnumberformat}%
1885 \edef\@glsl@counter{\csname glo@glsl@detoklabel{glsl@label}@counter\endcsname}%
1886 \glxtr@saveentrycounter
1887 \@do@wrglossary{\glxtr@alias{glsl@label}}%
1888 \endgroup
1889 \fi
1890 }

```

xtrindexaliased

```

1891 \newcommand{\@no@glxtrindexaliased}{%
1892 \PackageError{glossaries-extra}{\string\glxtrindexaliased\space
1893 not permitted outside definition of \string\glxtrsetaliasnoindex}%
1894 {}%
1895 }

```

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

```

1896 \let\glxtrindexaliased\@no@glxtrindexaliased

```

tDefaultGlsOpts Set the default options for \glslink etc.

```

1897 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
1898 \renewcommand*{\@glsl@setdefault@glslink@opts}{%
1899 \setkeys{glslink}{#1}%
1900 \@glxtrsetaliasnoindex
1901 }%
1902 }

```

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

```

1903 \newcommand*{\glxtrifindexing}[2]{%
1904 \ifKV@glslink@noindex #2\else #1\fi
1905 }

```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```

1906 \renewcommand*{\glswriteentry}[2]{%
1907 \glxtrifindexing
1908 {%
1909 \ifglslindexonlyfirst
1910 \ifglslused{#1}
1911 {\glxtrdoautoindexname{#1}{dualindex}}%

```

```

1912     {#2}%
1913   \else
1914     \glsifattribute{#1}{indexonlyfirst}{true}%
1915     {\ifglsused{#1}
1916      {\glsxtrdoautoindexname{#1}{dualindex}}}%
1917     {#2}}%
1918   {#2}%
1919   \fi
1920 }%
1921 {}%
1922 }

```

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

```

1923 \appto\@do@@wrglossary{\@glsxtr@do@@wrindex
1924   \glsxtrdowrglossaryhook{\@gls@label}}%
1925 }

```

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

Similarly for the “noidx” version:

`s@noidxglossary`

```

1926 \appto\gls@noidxglossary{\@glsxtr@do@@wrindex
1927   \glsxtrdowrglossaryhook{\@gls@label}}%
1928 }

```

`xtr@do@@wrindex`

```

1929 \newcommand*{\@glsxtr@do@@wrindex}{%
1930   \glsxtrdoautoindexname{\@gls@label}{dualindex}}%
1931 }

```

`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.)

```

1932 \newcommand*{\glsxtrdowrglossaryhook}[1]{%

```

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

```

1933 \newcommand*{\@gls@alt@hyp@opt}[1]{%
1934   \let\glslinkvar\@firstofthree
1935   \let\@gls@hyp@opt@cs#1\relax
1936   \@ifstar{\s@gls@hyp@opt}%
1937   {\@ifnextchar+%
1938    {\@firstoftwo{\p@gls@hyp@opt}}}%
1939   {%
1940     \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
1941     {\@firstoftwo{\@alt@gls@hyp@opt}}}%
1942     {#1}}%
1943 }%

```

```

1944 }%
1945 }

```

`\alt@glshyp@opt` User version

```

1946 \newcommand*{\@alt@glshyp@opt}[1][]{%
1947 \let\glslinkvar\@firstofthree
1948 \expandafter\@glshyp@opt@cs\expandafter[\@glshyp@opt@keys,#1]}

```

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

```

1949 \newcommand*{\@glshyp@opt@char}{}

```

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

```

1950 \newcommand*{\@glshyp@opt@keys}{}

```

`\rSetAltModifier`

```

1951 \newcommand*{\GlsXtrSetAltModifier}[2]{%
1952 \let\@glshyp@opt\@glshyp@opt
1953 \def\@glshyp@opt@char{#1}%
1954 \def\@glshyp@opt@keys{#2}%
1955 }

```

`\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 `\glstryshort` and, similarly, the long form commands like `\acrlong` and `\glsxtrlong` to use `\glstrylong`. Added attribute check.

```

1956 \renewcommand*{\glsdohyperlink}[2]{%
1957 \glshasattribute{\glslabel}{targeturl}%
1958 {%
1959 \glshasattribute{\glslabel}{targetname}%
1960 {%
1961 \glshasattribute{\glslabel}{targetcategory}%
1962 {%
1963 \hyperref{\glsggetattribute{\glslabel}{targeturl}}{%
1964 \glsggetattribute{\glslabel}{targetcategory}}%
1965 \glsggetattribute{\glslabel}{targetname}}%
1966 {\glsxtrprotectlinks#2}}%
1967 }%
1968 {%
1969 \hyperref{\glsggetattribute{\glslabel}{targeturl}}{%
1970 }%
1971 \glsggetattribute{\glslabel}{targetname}}%
1972 {\glsxtrprotectlinks#2}}%
1973 }%
1974 }%
1975 }%

```

```

1976 \href{\glsgetattribute{\glslabel}{targeturl}}%
1977 {{\glsxtrprotectlinks#2}}%
1978 }%
1979 }%
1980 {%

```

Check for alias.

```

1981 \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
1982 \ifdefined\gloaliaslabel
1983 {%
1984 \hyperlink{#1}{{\glsxtrprotectlinks#2}}%
1985 }%
1986 {%

```

Redirect link to the alias target.

```

1987 \hyperlink
1988 {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
1989 {{\glsxtrprotectlinks#2}}%
1990 }%
1991 }%
1992 }

```

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

```

1993 \ifundefined\glsdonohyperlink
1994 {%
1995 \renewcommand{\glsdisablehyper}{%
1996 \KV@glslink@hyperfalse
1997 \let\@glslink\glsdonohyperlink
1998 \let\@glstarget\@secondoftwo
1999 }
2000 }
2001 {}

```

`\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.

```

2002 \def\glsdonohyperlink#1#2{{\glsxtrprotectlinks #2}}

```

Reset `\@glslink` with patched versions:

```

2003 \ifcsundef{hyperlink}%
2004 {%
2005 \let\@glslink\glsdonohyperlink
2006 }%
2007 {%
2008 \let\@glslink\glsdohyperlink
2009 }

```

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

```

2010 \newcommand*{\glxtrprotectlinks}{%
2011   \KV@glslink@hyperfalse
2012   \KV@glslink@noindextrue
2013   \let\@gls@\@glxtrp@text@
2014   \let\@Gls@\@Glsxtrp@text@
2015   \let\@GLS@\@GLSxtrp@text@
2016   \let\@glspl@\@glxtrp@plural@
2017   \let\@Glspl@\@Glsxtrp@plural@
2018   \let\@GLSpl@\@GLSxtrp@plural@
2019   \let\@glxtrshort@\@glxtrp@short@
2020   \let\@Glsxtrshort@\@Glsxtrp@short@
2021   \let\@GLSxtrshort@\@GLSxtrp@short@
2022   \let\@glxtrlong@\@glxtrp@long@
2023   \let\@Glsxtrlong@\@Glsxtrp@long@
2024   \let\@GLSxtrlong@\@GLSxtrp@long@
2025   \let\@glxtrshortpl@\@glxtrp@shortpl@
2026   \let\@Glsxtrshortpl@\@Glsxtrp@shortpl@
2027   \let\@GLSxtrshortpl@\@GLSxtrp@shortpl@
2028   \let\@glxtrlongpl@\@glxtrp@longpl@
2029   \let\@Glsxtrlongpl@\@Glsxtrp@longpl@
2030   \let\@GLSxtrlongpl@\@GLSxtrp@longpl@
2031   \let\@acrshort@\@glxtrp@acrshort@
2032   \let\@Acrshort@\@Glsxtrp@acrshort@
2033   \let\@ACRshort@\@GLSxtrp@acrshort@
2034   \let\@acrshortpl@\@glxtrp@acrshortpl@
2035   \let\@Acrshortpl@\@Glsxtrp@acrshortpl@
2036   \let\@ACRshortpl@\@GLSxtrp@acrshortpl@
2037   \let\@acrlong@\@glxtrp@acrlong@
2038   \let\@Acrlong@\@Glsxtrp@acrlong@
2039   \let\@ACRlong@\@GLSxtrp@acrlong@
2040   \let\@acrlongpl@\@glxtrp@acrlongpl@
2041   \let\@Acrlongpl@\@Glsxtrp@acrlongpl@
2042   \let\@ACRlongpl@\@GLSxtrp@acrlongpl@
2043 }

```

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

@glxtrp@text@

```
2044 \def\@glxtrp@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}
```

@Glsxtrp@text@

```
2045 \def\@Glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtrp@text@

```
2046 \def\@GLSxtrp@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2047 \def\@glxtrp@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

lsxtr@p@plural@

```
2048 \def\@Glsxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

LSxtr@p@plural@

```
2049 \def\@GLSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glxtr@p@short@

```
2050 \def\@glxtr@p@short@#1#2[#3]{%
2051  {%
2052   \glsssetabbrvfmt{\glscategory{#2}}%
2053   \glsabbrvfont{\glentryshort{#2}}#3%
2054  }%
2055 }
```

Glsxtr@p@short@

```
2056 \def\@Glsxtr@p@short@#1#2[#3]{%
2057  {%
2058   \glsssetabbrvfmt{\glscategory{#2}}%
2059   \glsabbrvfont{\Glsentryshort{#2}}#3%
2060  }%
2061 }
```

GLSxtr@p@short@

```
2062 \def\@GLSxtr@p@short@#1#2[#3]{%
2063  {%
2064   \glsssetabbrvfmt{\glscategory{#2}}%
2065   \mfirstucMakeUppercase{\glsabbrvfont{\glentryshort{#2}}#3}%
2066  }%
2067 }
```

sxtr@p@shortpl@

```
2068 \def\@glxtr@p@shortpl@#1#2[#3]{%
2069  {%
2070   \glsssetabbrvfmt{\glscategory{#2}}%
2071   \glsabbrvfont{\glentryshortpl{#2}}#3%
2072  }%
2073 }
```

sxtr@p@shortpl@

```
2074 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2075  {%
2076   \glsssetabbrvfmt{\glscategory{#2}}%
2077   \glsabbrvfont{\Glsentryshortpl{#2}}#3%
2078  }%
2079 }
```

Sxtr@p@shortpl@

```
2080 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
```

```

2081  {%
2082    \glsssetabbrvfmt{\glscategory{#2}}%
2083    \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshortpl{#2}}#3}%
2084  }%
2085 }

@glxstr@p@long@
2086 \def\@glxstr@p@long@#1#2[#3]{\glsentrylong{#2}#3}}

@Glxstr@p@long@
2087 \def\@Glxstr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}}

@GLSxtr@p@long@
2088 \def\@GLSxtr@p@long@#1#2[#3]{%
2089   {\mfirstucMakeUppercase{\glslongfont{\glsentrylong{#2}}#3}}}

lsxtr@p@longpl@
2090 \def\@lsxtr@p@longpl@#1#2[#3]{\glsentrylongpl{#2}#3}}

lsxtr@p@longpl@
2091 \def\@Lsxtr@p@longpl@#1#2[#3]{\glslongfont{\Glsentrylongpl{#2}}#3}}

LSxtr@p@longpl@
2092 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2093   {\mfirstucMakeUppercase{\glslongfont{\glsentrylongpl{#2}}#3}}}

xtr@p@acrshort@
2094 \def\@gxstr@p@acrshort@#1#2[#3]{\acronymfont{\glsentryshort{#2}}#3}}

xtr@p@acrshort@
2095 \def\@Glxstr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
2096 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2097   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
2098 \def\@glxstr@p@acrshortpl@#1#2[#3]{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2099 \def\@Glxstr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2100 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2101   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}

sxtr@p@acrlong@
2102 \def\@glxstr@p@acrlong@#1#2[#3]{\glsentrylong{#2}#3}}

```

```

sxt@p@acrlong@
2103 \def\@Glsxt@p@acrlong@#1#2[#3]{\Glsentrylong{#2}#3}}

Sxt@p@acrlong@
2104 \def\@GLSxt@p@acrlong@#1#2[#3]{%
2105 {\mfirstucMakeUppercase{\Glsentrylong{#2}#3}}}}

tr@p@acrlongpl@
2106 \def\@Glsxt@p@acrlongpl@#1#2[#3]{\Glsentrylongpl{#2}#3}}

tr@p@acrlongpl@
2107 \def\@Glsxt@p@acrlongpl@#1#2[#3]{\Glsentrylongpl{#2}#3}}

tr@p@acrlongpl@
2108 \def\@GLSxt@p@acrlongpl@#1#2[#3]{%
2109 {\mfirstucMakeUppercase{\Glsentrylongpl{#2}#3}}}}

```

Commands to minimise conflict.

```

\@glsxtrp@opt
2110 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

\glsxtrsetpopts  Used in glossary to switch hyperlinks on for the \glsxtrp type of commands.
2111 \newcommand*{\glsxtrsetpopts}[1]{%
2112 \renewcommand*{\@glsxtrp@opt}{#1}%
2113 }

lossxtrsetpopts  Used in glossary to switch hyperlinks on for the \glsxtrp type of commands.
2114 \newcommand*{\glossxtrsetpopts}{%
2115 \glsxtrsetpopts{noindex}%
2116 }

\@@glsxtrp
2117 \newrobustcmd*{\@@glsxtrp}[2]{%
  Add scope.
2118 {%
2119 \let\glspostlinkhook\relax
2120 \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
2121 }%
2122 }

\@glsxtrp
2123 \newrobustcmd*{\@glsxtrp}[2]{%
2124 \ifcsdef{gls#1}%
2125 {%
2126 \@glsxtrp{gls#1}{#2}%
2127 }%

```



```

2128 {%
2129   \ifcsdef{glsxtr#1}%
2130   {%
2131     \@glsxtrp{glsxtr#1}{#2}%
2132   }%
2133   {%
2134     \PackageError{glossaries-extra}{‘#1’ not recognised by
2135       \string\glsxtrp}{}%
2136   }%
2137 }%
2138 }

```

\@Glsxtrp

```

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

```

\@GLSxtrp

```

2155 \newrobustcmd*{\@GLSxtrp}[2]{%
2156   \ifcsdef{GLS#1}%
2157   {%
2158     \@glsxtrp{GLS#1}{#2}%
2159   }%
2160   {%
2161     \ifcsdef{GLSxtr#1}%
2162     {%
2163       \@glsxtrp{GLSxtr#1}{#2}%
2164     }%
2165     {%
2166       \PackageError{glossaries-extra}{‘#1’ not recognised by
2167         \string\GLSxtrp}{}%
2168     }%
2169   }%
2170 }

```

\glsxtr@entry@p

```

2171 \newrobustcmd*{\glsxtr@headentry@p}[2]{%
2172   \glsifattribute{#1}{headuc}{true}%
2173   {%
2174     \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
2175   }%
2176   {%
2177     \@gls@entry@field{#1}{#2}%
2178   }%
2179 }

```

`\glsxtrp` Not robust as it needs to expand somewhat.

```

2180 \ifdef\teorpdfstring
2181 {
2182   \newcommand{\glsxtrp}[2]{%
2183     \protect\NoCaseChange
2184     {%
2185       \protect\teorpdfstring
2186       {%
2187         \protect\glsxtrifinmark
2188         {%
2189           \ifcsdef{glsxtrhead#1}%
2190           {%
2191             {\protect\csuse{glsxtrhead#1}{#2}}%
2192           }%
2193           {%
2194             \glsxtr@headentry@p{#2}{#1}%
2195           }%
2196         }%
2197         {%
2198           \@glsxtrp{#1}{#2}%
2199         }%
2200       }%
2201       {%
2202         \protect\@gls@entry@field{#2}{#1}%
2203       }%
2204     }%
2205   }
2206 }
2207 {
2208   \newcommand{\glsxtrp}[2]{%
2209     \protect\NoCaseChange
2210     {%
2211       \protect\glsxtrifinmark
2212       {%
2213         \ifcsdef{glsxtrhead#1}%
2214         {%
2215           {\protect\csuse{glsxtrhead#1}}%
2216         }%
2217         {%

```

```

2218      \glxtr@headentry@p{#2}{#1}%
2219    }%
2220  }%
2221  {%
2222    \@glxtrp{#1}{#2}%
2223  }%
2224  }%
2225 }
2226 }

```

Provide short synonyms for the most common option.

\glsp

```

2227 \newcommand*{\glsp}{\glxtrp{short}}

```

\glsp

```

2228 \newcommand*{\glsp}{\glxtrp{text}}

```

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

```

2229 \ifdef\teorpdfstring
2230 {
2231   \newcommand{\Glsxtrp}[2]{%
2232     \protect\NoCaseChange
2233     {%
2234       \protect\teorpdfstring
2235       {%
2236         \protect\glxtrifinmark
2237         {%
2238           \ifcsdef{Glsxtrhead#1}%
2239             {%
2240               {\protect\csuse{Glsxtrhead#1}{#2}}%
2241             }%
2242             {%
2243               \protect\@Gls@entry@field{#2}{#1}%
2244             }%
2245           }%
2246           {%
2247             \@Glsxtrp{#1}{#2}%
2248           }%
2249         }%
2250         {%
2251           \protect\@gl@entry@field{#2}{#1}%
2252         }%
2253       }%
2254     }
2255 }
2256 {
2257   \newcommand{\Glsxtrp}[2]{%

```

```

2258 \protect\NoCaseChange
2259 {%
2260 \protect\glxtrifinmark
2261 {%
2262 \ifcsdef{Glsxtrhead#1}%
2263 {%
2264 {\protect\csuse{Glsxtrhead#1}}%
2265 }%
2266 {%
2267 \protect\@Gls@entry@field{#2}{#1}%
2268 }%
2269 }%
2270 {%
2271 \@Glsxtrp{#1}{#2}%
2272 }%
2273 }%
2274 }
2275 }

```

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

```

2276 \ifdef\texorpdfstring
2277 {
2278 \newcommand{\GLSxtrp}[2]{%
2279 \protect\NoCaseChange
2280 {%
2281 \protect\texorpdfstring
2282 {%
2283 \protect\glxtrifinmark
2284 {%
2285 \ifcsdef{GLSxtr#1}%
2286 {%
2287 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2288 }%
2289 {%
2290 \protect\mfirstucMakeUppercase
2291 {%
2292 \protect\@gls@entry@field{#2}{#1}%
2293 }%
2294 }%
2295 }%
2296 {%
2297 \@GLSxtrp{#1}{#2}%
2298 }%
2299 }%
2300 {%
2301 \protect\@gls@entry@field{#2}{#1}%
2302 }%
2303 }%
2304 }

```

```

2305 }
2306 {
2307   \newcommand{\GLSxtrp}[2]{%
2308     \protect\NoCaseChange
2309     {%
2310       \protect\glsxtrifinmark
2311       {%
2312         \ifcsdef{GLSxtr#1}%
2313         {%
2314           {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2315         }%
2316         {%
2317           \protect\mfirstucMakeUppercase
2318           {%
2319             \protect\@gls@entry@field{#2}{#1}%
2320           }%
2321         }%
2322       }%
2323     }%
2324     \@GLSxtrp{#1}{#2}%
2325   }%
2326 }%
2327 }
2328 }

```

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 instead.

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

\@glsunset Global unset.

```

2329 \renewcommand*{\@glsunset}[1]{%
2330   \@glsunset{#1}%
2331   \glsxtrpostunset{#1}%
2332 }%

```

glsxtrpostunset

```

2333 \newcommand*{\glsxtrpostunset}[1]{}

```

\@glslocalunset Local unset.

```

2334 \renewcommand*{\@glslocalunset}[1]{%
2335   \@glslocalunset{#1}%
2336   \glsxtrpostlocalunset{#1}%
2337 }%

```

rpostlocalunset

```

2338 \newcommand*{\glsxtrpostlocalunset}[1]{}

```

```

\@glsreset   Global reset.
2339 \renewcommand*{\@glsreset}[1]{%
2340   \@glsreset{#1}%
2341   \glsxtrpostreset{#1}%
2342 }%

glsxtrpostreset
2343 \newcommand*{\glsxtrpostreset}[1]{%

\@glslocalreset   Local reset.
2344 \renewcommand*{\@glslocalreset}[1]{%
2345   \@glslocalreset{#1}%
2346   \glsxtrpostlocalreset{#1}%
2347 }%

rpostlocalreset
2348 \newcommand*{\glsxtrpostlocalreset}[1]{%

leEntryCounting   The first argument is the list of categories and the second argument is the value of the en-
                    trycount attribute.
2349 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
    Enable entry counting:
2350   \glsenableentrycount
    Redefine \gls etc:
2351   \renewcommand*{\gls}{\cglsl}%
2352   \renewcommand*{\Gls}{\cGls}%
2353   \renewcommand*{\glspl}{\cglspl}%
2354   \renewcommand*{\Glspl}{\cGlspl}%
2355   \renewcommand*{\GLS}{\cGLS}%
2356   \renewcommand*{\GLSpl}{\cGLSpl}%
    Set the entrycount attribute:
2357   \@glsxtr@setentrycountunsetattr{#1}{#2}%
    In case this command is used again:
2358   \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
2359   \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
2360     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2361       can't be used with \string\GlsXtrEnableEntryCounting}%
2362     {Use one or other but not both commands}}%
2363 }

ycountunsetattr
2364 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
2365   \@for\@glsxtr@cat:=#1\do
2366   {%
2367     \ifdefempty{\@glsxtr@cat}{}%
2368     {%

```

```

2369     \glssetcategoryattribute{\@glstr@cat}{entrycount}{#2}%
2370 }%
2371 }%
2372 }

```

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

nableentrycount

```

2373 \renewcommand*\glsenableentrycount{%
  Enable new fields:
2374   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
  Just in case the user has switched on the docdef option.
2375   \renewcommand*\gls@defdocnewglossaryentry{%
2376     \renewcommand*\newglossaryentry[2]{%
2377       \PackageError{glossaries}{\string\newglossaryentry\space
2378         may only be used in the preamble when entry counting has
2379         been activated}{If you use \string\glsenableentrycount\space
2380         you must place all entry definitions in the preamble not in
2381         the document environment}%
2382     }%
2383 }%

```

New commands to access new fields:

```

2384 \newcommand*\glstentrycurrcount[1]{%
2385   \ifcsundef{glo@\glsdetoklabel{##1}@currcount}%
2386   {0}{\@gls@entry@field{##1}{currcount}}%
2387 }%
2388 \newcommand*\glstentryprevcount[1]{%
2389   \ifcsundef{glo@\glsdetoklabel{##1}@prevcount}%
2390   {0}{\@gls@entry@field{##1}{prevcount}}%
2391 }%

```

Adjust post unset and reset:

```

2392 \let\@glstr@entrycount@org@unset\glstrpostunset
2393 \renewcommand*\glstrpostunset[1]{%
2394   \@glstr@entrycount@org@unset{##1}%
2395   \@gls@increment@currcount{##1}%
2396 }%
2397 \let\@glstr@entrycount@org@localunset\glstrpostlocalunset
2398 \renewcommand*\glstrpostlocalunset[1]{%
2399   \@glstr@entrycount@org@localunset{##1}%
2400   \@gls@local@increment@currcount{##1}%
2401 }%
2402 \let\@glstr@entrycount@org@reset\glstrpostreset
2403 \renewcommand*\glstrpostreset[1]{%
2404   \@glstr@entrycount@org@reset{##1}%
2405   \csgdef{glo@\glsdetoklabel{##1}@currcount}{0}%
2406 }%
2407 \let\@glstr@entrycount@org@localreset\glstrpostlocalreset

```

```

2408 \renewcommand*{\glxstrpostlocalreset}[1]{%
2409   \@glxstr@entrycount@org@localreset{##1}%
2410   \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
2411 }%

```

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

```

2412 \let\@cgl@s\@cgl@s@
2413 \let\@cgl@sp1\@cgl@sp1@
2414 \let\@cGLS\@cGLS@
2415 \let\@cGL@sp1\@cGL@sp1@
2416 \let\@cGLS\@cGLS@
2417 \let\@cGL@sp1\@cGL@sp1@

```

The rest is as the original definition.

```

2418 \AtEndDocument{\@gl@s@write@entrycounts}%
2419 \renewcommand*{\@gl@s@entry@count}[2]{%
2420   \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
2421 }%
2422 \let\gl@enable@entrycount\relax
2423 \renewcommand*{\gl@enable@entryunitcount}{%
2424   \PackageError{glossaries-extra}{\string\gl@enable@entryunitcount\space
2425     can't be used with \string\gl@enable@entrycount}%
2426   {Use one or other but not both commands}%
2427 }%
2428 }

```

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

```

2429 \renewcommand*{\@gl@s@write@entrycounts}{%
2430   \immediate\write\@auxout
2431     {\string\providecommand*{\string\@gl@s@entry@count}[2]{}}%
2432   \count@=0\relax
2433   \forallgl@sentries{\@gl@sentry}{%
2434     \gl@hasattribute{\@gl@sentry}{entrycount}%
2435     {%
2436       \ifgl@sused{\@gl@sentry}%
2437       {%
2438         \immediate\write\@auxout
2439           {\string\@gl@s@entry@count{\@gl@sentry}{\gl@sentrycurrcount{\@gl@sentry}}}%
2440       }%
2441     }%
2442     \advance\count@ by \@ne
2443   }%
2444 }%
2445 }%
2446 \ifnum\count@=0
2447   \GlossariesExtraWarningNoLine{Entry counting has been enabled
2448     \MessageBreak with \string\gl@enable@entrycount\space but the
2449     \MessageBreak attribute 'entrycount' hasn't

```



```

2450     \MessageBreak been assigned to any of the defined
2451     \MessageBreak entries}%
2452 \fi
2453 }

```

trifcounttrigger

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

```

2454 \newcommand*{\glxtrifcounttrigger}[3]{%
2455   \glshasattribute{#1}{entrycount}%
2456   {%
2457     \ifnum\gl Sentryprevcount{#1}>\gl sgetattribute{#1}{entrycount}\relax
2458     #3%
2459   \else
2460     #2%
2461   \fi
2462 }%
2463 {#3}%
2464 }

```

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

\@@cgl@

```

2465 \def\@@cgl@#1#2[#3]{%
2466   \glxtrifcounttrigger{#2}%
2467   {%
2468     \cgl sformat{#2}{#3}%
2469     \gl sunset{#2}%
2470   }%
2471   {%
2472     \@gls@{#1}{#2}[#3]%
2473   }%
2474 }%

```

\@@cgl@

```

2475 \def\@@cgl spl@#1#2[#3]{%
2476   \glxtrifcounttrigger{#2}%
2477   {%
2478     \cgl splformat{#2}{#3}%
2479     \gl sunset{#2}%
2480   }%
2481   {%
2482     \@gls pl@{#1}{#2}[#3]%
2483   }%
2484 }%

```

\@@cGls@

```
2485 \def\@@cGls@#1#2[#3]{%
2486   \glstrifcounttrigger{#2}%
2487   {%
2488     \cGlsformat{#2}{#3}%
2489     \glset{#2}%
2490   }%
2491   {%
2492     \@Gls@{#1}{#2}[#3]%
2493   }%
2494 }%
```

\@@cGlspl@

```
2495 \def\@@cGlspl@#1#2[#3]{%
2496   \glstrifcounttrigger{#2}%
2497   {%
2498     \cGlsplformat{#2}{#3}%
2499     \glset{#2}%
2500   }%
2501   {%
2502     \@Glspl@{#1}{#2}[#3]%
2503   }%
2504 }%
```

\@@cGLS@

```
2505 \def\@@cGLS@#1#2[#3]{%
2506   \glstrifcounttrigger{#2}%
2507   {%
2508     \cGLSformat{#2}{#3}%
2509     \glset{#2}%
2510   }%
2511   {%
2512     \@GLS@{#1}{#2}[#3]%
2513   }%
2514 }%
```

\@@cGLSpl@

```
2515 \def\@@cGLSpl@#1#2[#3]{%
2516   \glstrifcounttrigger{#2}%
2517   {%
2518     \cGLSplformat{#2}{#3}%
2519     \glset{#2}%
2520   }%
2521   {%
2522     \@GLSpl@{#1}{#2}[#3]%
2523   }%
2524 }%
```

Remove default warnings from \cglS etc so that it can be used interchangeable with \glS etc.

```
\@cgl@s@
2525 \def\@cgl@s@#1#2[#3]{\@gls@{#1}-{#2} [#3]}
```

```
\@cGls@
2526 \def\@cGls@#1#2[#3]{\@Gls@{#1}-{#2} [#3]}
```

```
\@cgl spl@
2527 \def\@cgl spl@#1#2[#3]{\@glspl@{#1}-{#2} [#3]}
```

```
\@cGlspl@
2528 \def\@cGlspl@#1#2[#3]{\@Glspl@{#1}-{#2} [#3]}
```

Add all upper case versions not provided by glossaries.

```
\cGLS
2529 \newrobustcmd*{\cGLS}{\@gls@hyp@opt\@cGLS}
```

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

```
2530 \newcommand*{\@cGLS}[2][\%
2531 \new@ifnextchar[\@cGLS@{#1}-{#2}]{\@cGLS@{#1}-{#2} []}%
2532 }
```

```
\@cGLS@
2533 \def\@cGLS@#1#2[#3]{\@Gls@{#1}-{#2} [#3]}
```

\cGLSformat Format used by \cGLS if entry only used once on previous run. The first argument is the label, the second argument is the insert text.

```
2534 \newcommand*{\cGLSformat}[2]{\%
2535 \expandafter\mfirstucMakeUppercase\expandafter{\cgl sformat{#1}-{#2}}}%
2536 }
```

```
\cGLSpl
2537 \newrobustcmd*{\cGLSpl}{\@gls@hyp@opt\@cGLSpl}
```

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

```
2538 \newcommand*{\@cGLSpl}[2][\%
2539 \new@ifnextchar[\@cGLSpl@{#1}-{#2}]{\@cGLSpl@{#1}-{#2} []}%
2540 }
```

```
\@cGLSpl@
2541 \def\@cGLSpl@#1#2[#3]{\@Glspl@{#1}-{#2} [#3]}
```

\cGLSplformat Format used by \cGLSpl if entry only used once on previous run. The first argument is the label, the second argument is the insert text.

```
2542 \newcommand*{\cGLSplformat}[2]{\%
2543 \expandafter\mfirstucMakeUppercase\expandafter{\cgl splformat{#1}-{#2}}}%
2544 }
```

Modify the trigger formats to check for the regular attribute.

`\cglformat`

```
2545 \renewcommand*{\cglformat}[2]{%
2546   \glsifregular{#1}
2547   {\glsentryfirst{#1}}%
2548   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}#2%
2549 }
```

`\cGlsformat`

```
2550 \renewcommand*{\cGlsformat}[2]{%
2551   \glsifregular{#1}
2552   {\Glsentryfirst{#1}}%
2553   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}#2%
2554 }
```

`\cglsplformat`

```
2555 \renewcommand*{\cglsplformat}[2]{%
2556   \glsifregular{#1}
2557   {\glsentryfirstplural{#1}}%
2558   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}#2%
2559 }
```

`\cGlsplformat`

```
2560 \renewcommand*{\cGlsplformat}[2]{%
2561   \glsifregular{#1}
2562   {\Glsentryfirstplural{#1}}%
2563   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2%
2564 }
```

New code similar to above for unit counting.

`defunitcounters`

```
2565 \newcommand*{\@@newglossaryentry@defunitcounters}{%
2566   \edef\@glo@countunit{\csuse{@glsxtr@categoryattr@@\@glo@category @unitcount}}%
2567   \ifdefvoid\@glo@countunit
2568   {}%
2569   {%
2570     \@glsxtr@ifunitcounter{\@glo@countunit}%
2571     {}%
2572     {\expandafter\@glsxtr@addunitcounter\expandafter{\@glo@countunit}}%
2573   }%
2574 }
```

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

```
2575 \newcommand*{\@glsxtr@unitcountlist}{}
```

@addunitcounter

```
2576 \newcommand*{\@glxstr@addunitcounter}[1]{%
2577 \listadd{\@glxstr@unitcountlist}{#1}%
2578 \ifcsundef{glxstr@theunit@#1}
2579 {%
2580 \ifcsdef{theH#1}%
2581 {\csdef{glxstr@theunit@#1}{\csuse{theH#1}}}%
2582 {\csdef{glxstr@theunit@#1}{\csuse{the#1}}}%
2583 }%
2584 {}%
2585 }
```

r@ifunitcounter

```
2586 \newcommand*{\@glxstr@ifunitcounter}[3]{%
2587 \xifinlist{#1}{\@glxstr@unitcountlist}{#2}{#3}%
2588 }
```

urrentunitcount

```
2589 \newcommand*\@glxstr@currentunitcount[1]{%
2590 glo@\glsdetoklabel{#1}@currunit@glsggetattribute{#1}{unitcount}.%
2591 \csuse{glxstr@theunit@glsggetattribute{#1}{unitcount}}%
2592 }
```

eviousunitcount

```
2593 \newcommand*\@glxstr@previousunitcount[1]{%
2594 glo@\glsdetoklabel{#1}@prevunit@glsggetattribute{#1}{unitcount}.%
2595 \csuse{glxstr@theunit@glsggetattribute{#1}{unitcount}}%
2596 }
```

t@currunitcount

```
2597 \newcommand*{\@glx@increment@currunitcount}[1]{%
2598 \glshasattribute{#1}{unitcount}%
2599 {%
2600 \edef\@glxstr@csname{\@glxstr@currentunitcount{#1}}%
2601 \ifcsundef{\@glxstr@csname}%
2602 {%
2603 \csgdef{\@glxstr@csname}{1}%
2604 \listcsxadd
2605 {glo@\glsdetoklabel{#1}@unitlist}%
2606 {\glsggetattribute{#1}{unitcount}.%
2607 \csuse{glxstr@theunit@glsggetattribute{#1}{unitcount}}}%
2608 }%
2609 }%
2610 {%
2611 \csxdef{\@glxstr@csname}%
2612 {\number\numexpr\csname\@glxstr@csname\endcsname+1}%
2613 }%
2614 }%
2615 {}%
```

```

2616 }

t@currunitcount
2617 \newcommand*{\@gls@local@increment@currunitcount}[1]{%
2618   \gls@hasattribute{#1}{unitcount}%
2619   {%
2620     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
2621     \ifcsundef{\@glsxtr@csname}%
2622     {%
2623       \csdef{\@glsxtr@csname}{1}%
2624       \listcseadd
2625         {glo\@glsdetoklabel{#1}@unitlist}%
2626         {\glsgetattribute{#1}{unitcount}.%
2627          \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
2628       }%
2629     }%
2630   {%
2631     \csedef{\@glsxtr@csname}%
2632       {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
2633     }%
2634   }%
2635   {}%
2636 }

r@currunitcount
2637 \newcommand*{\@glsxtr@currunitcount}[2]{%
2638   \ifcsundef
2639     {glo\@glsdetoklabel{#1}@currunit@#2}%
2640     {0}%
2641     {\csuse{glo\@glsdetoklabel{#1}@currunit@#2}}%
2642   }%

r@prevunitcount
2643 \newcommand*{\@glsxtr@prevunitcount}[2]{%
2644   \ifcsundef
2645     {glo\@glsdetoklabel{#1}@prevunit@#2}%
2646     {0}%
2647     {\csuse{glo\@glsdetoklabel{#1}@prevunit@#2}}%
2648   }%

eentryunitcount
2649 \newcommand*{\@glsenableentryunitcount}{%
  Enable new fields:
2650   \appto\@newglossaryentry@defcounters{\@@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
2651   \renewcommand*{\@gls@defdocnewglossaryentry}{%
2652     \renewcommand*\newglossaryentry[2]{%
2653       \PackageError{glossaries}{\string\newglossaryentry\space

```

```

2654     may only be used in the preamble when entry counting has
2655     been activated}{If you use \string\glsenableentryunitcount\space
2656     you must place all entry definitions in the preamble not in
2657     the document environment}%
2658 }%
2659 }%

```

New commands to access new fields:

```

2660 \newcommand*{\glsentrycurrcount}[1]{%
2661   \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2662   \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2663 }%
2664 \newcommand*{\glsentryprevcount}[1]{%
2665   \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2666   \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2667 }%

```

Access total count:

```

2668 \newcommand*{\glsentryprevtotalcount}[1]{%
2669   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
2670   {0}%
2671   {%
2672     \number\csuse{glo@\glsdetoklabel{##1}@prevunittotal}
2673   }%
2674 }%

```

Access max value:

```

2675 \newcommand*{\glsentryprevmaxcount}[1]{%
2676   \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
2677   {0}%
2678   {%
2679     \number\csuse{glo@\glsdetoklabel{##1}@prevunitmax}
2680   }%
2681 }%

```

Adjust post unset and reset:

```

2682 \let\@glsxtr@entryunitcount@org@unset\glsxtrpostunset
2683 \renewcommand*{\glsxtrpostunset}[1]{%
2684   \@glsxtr@entryunitcount@org@unset{##1}%
2685   \@gls@increment@currunitcount{##1}%
2686 }%
2687 \let\@glsxtr@entryunitcount@org@localunset\glsxtrpostlocalunset
2688 \renewcommand*{\glsxtrpostlocalunset}[1]{%
2689   \@glsxtr@entryunitcount@org@localunset{##1}%
2690   \@gls@local@increment@currunitcount{##1}%
2691 }%
2692 \let\@glsxtr@entryunitcount@org@reset\glsxtrpostreset
2693 \renewcommand*{\glsxtrpostreset}[1]{%
2694   \glshasattribute{##1}{unitcount}%
2695   {%
2696     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{##1}}%

```

```

2697     \ifcsundef{\@glxtr@csname}%
2698     {}%
2699     {\csgdef{\@glxtr@csname}{0}}%
2700 }%
2701 {}%
2702 }%
2703 \let\@glxtr@entryunitcount@org@localreset\glxtrpostlocalreset
2704 \renewcommand*{\glxtrpostlocalreset}[1]{%
2705   \@glxtr@entryunitcount@org@localreset{##1}%
2706   \glshasattribute{##1}{unitcount}%
2707   {%
2708     \edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
2709     \ifcsundef{\@glxtr@csname}%
2710     {}%
2711     {\csdef{\@glxtr@csname}{0}}%
2712   }%
2713   {}%
2714 }%

```

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

```

2715 \let\@cgl@%cgl%
2716 \let\@cglsp@%cglsp%
2717 \let\@cGL@%cGL%
2718 \let\@cGLsp@%cGLsp%
2719 \let\@cGLS@%cGLS%
2720 \let\@cGLSp@%cGLSp%

```

Write information to the aux file.

```

2721 \AtEndDocument{\@gls@write@entryunitcounts}%
2722 \renewcommand*{\@gls@entry@unitcount}[3]{%
2723   \csgdef{glo@glstdetoklabel{##1}@prevunit@##3}{##2}%
2724   \ifcsundef{glo@glstdetoklabel{##1}@prevunittotal}%
2725   {\csgdef{glo@glstdetoklabel{##1}@prevunittotal}{##2}}%
2726   {%
2727     \csxdef{glo@glstdetoklabel{##1}@prevunittotal}{
2728       \number\numexpr\csuse{glo@glstdetoklabel{##1}@prevunittotal}+##2}%
2729     }%
2730   \ifcsundef{glo@glstdetoklabel{##1}@prevunitmax}%
2731   {\csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}}%
2732   {%
2733     \ifnum\csuse{glo@glstdetoklabel{##1}@prevunitmax}<##2
2734     \csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}%
2735     \fi
2736   }%
2737 }%
2738 \let\glsenableentryunitcount\relax
2739 \renewcommand*{\glsenableentrycount}{%
2740   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
2741     can't be used with \string\glsenableentryunitcount}%

```



```

2742     {Use one or other but not both commands}%
2743 }%
2744 }
2745 \onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

2746 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

2747 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
2748   \immediate\write\@auxout
2749   {\string\@gls@entry@unitcount
2750    \@glsentry}%
2751   {\@glsxtr@currunitcount{\@glsentry}{#1}%
2752   }%
2753   {#1}}%
2754 }

```

entryunitcounts

```

2755 \newcommand*{\@gls@write@entryunitcounts}{%
2756   \immediate\write\@auxout
2757   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
2758   \count@=0\relax
2759   \forallglsentries{\@glsentry}{%
2760     \glshasattribute{\@glsentry}{unitcount}%
2761     {%
2762       \ifglsused{\@glsentry}%
2763       {%
2764         \forlistcsloop
2765           {\@gls@write@entryunitcounts@do}%
2766           {glo@\@glsdetoklabel{\@glsentry}@unitlist}%
2767       }%
2768     }%
2769     \advance\count@ by \@ne
2770   }%
2771   {}%
2772 }%
2773 \ifnum\count@=0
2774   \GlossariesExtraWarningNoLine{Entry counting has been enabled
2775   \MessageBreak with \string\glsenableentryunitcount\space but the
2776   \MessageBreak attribute ‘unitcount’ hasn’t
2777   \MessageBreak been assigned to any of the defined
2778   \MessageBreak entries}%
2779 \fi
2780 }

```

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.

```

2781 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```
2782 \glsenableentryunitcount
```

Redefine \gls etc:

```
2783 \renewcommand*{\gls}{\cglsl}%
2784 \renewcommand*{\Gls}{\cGls}%
2785 \renewcommand*{\glspl}{\cglspl}%
2786 \renewcommand*{\Glspl}{\cGlspl}%
2787 \renewcommand*{\GLS}{\cGLS}%
2788 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2789 \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%
```

In case this command is used again:

```
2790 \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
2791 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[2]{%
2792   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2793     can't be used with \string\GlsXtrEnableEntryUnitCounting}%
2794   {Use one or other but not both commands}}%
2795 }
```

countunsetattr

```
2796 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
2797   \@for\@glsxtr@cat:=#1\do
2798   {%
2799     \ifdefempty{\@glsxtr@cat}{}%
2800     {%
2801       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
2802       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
2803     }%
2804   }%
2805 }
```

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.

genericNewAcronym

```
2806 \renewcommand*{\SetGenericNewAcronym}{%
2807   \let\@Gls@entryname\@Gls@acrenryname
2808   \renewcommand{\newacronym}[4][ ]{%
2809     \ifdefempty{\@glsacronymlists}%
2810     {%
```

```

2811 \def\@glo@type{\acronymtype}%
2812 \setkeys{glossentry}{##1}%
2813 \DeclareAcronymList{\@glo@type}%
2814 }%
2815 {}%
2816 \glskeylisttok{##1}%
2817 \glslabeltok{##2}%
2818 \glsshorttok{##3}%
2819 \glslongtok{##4}%
2820 \newacronymhook
2821 \protected@edef\@do@newglossaryentry{%
2822 \noexpand\newglossaryentry{\the\glslabeltok}%
2823 {%
2824 type=\acronymtype,%
2825 name={\expandonce{\acronymentry{##2}}},%
2826 sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
2827 text={\the\glsshorttok},%
2828 short={\the\glsshorttok},%
2829 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
2830 long={\the\glslongtok},%
2831 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
2832 category=acronym,
2833 \GenericAcronymFields,%
2834 \the\glskeylisttok
2835 }%
2836 }%
2837 \@do@newglossaryentry
2838 }%
2839 \renewcommand*{\acrfullfmt}[3]{%
2840 \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
2841 \renewcommand*{\Acrfullfmt}[3]{%
2842 \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
2843 \renewcommand*{\ACRfullfmt}[3]{%
2844 \glslink[##1]{##2}{%
2845 \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}}%
2846 \renewcommand*{\acrfullplfmt}[3]{%
2847 \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
2848 \renewcommand*{\Acrfullplfmt}[3]{%
2849 \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
2850 \renewcommand*{\ACRfullplfmt}[3]{%
2851 \glslink[##1]{##2}{%
2852 \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}}%
2853 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
2854 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
2855 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
2856 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
2857 }

```

This will cause a problem for glossaries that contain a mixture of acronyms and abbrevia-

tions, so redefine `\newacronym` to use the new abbreviation interface.

First save the original definitions:

```
2858 \let\@glxtr@org@setacronymstyle\setacronymstyle
2859 \let\@glxtr@org@newacronymstyle\newacronymstyle
```

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

```
2860 \newcommand*{\MakeAcronymsAbbreviations}{%
2861   \renewcommand*{\newacronym}[4][]{%
2862     \glxtr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
2863   }%
2864   \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
2865   \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%
2866   \renewcommand*{\setacronymstyle}[1]{%
2867     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}
2868       unavailable.
2869     Use \string\setabbreviationstyle\space instead.
2870     The original acronym interface can be restored with
2871     \string\RestoreAcronyms}{}%
2872   }%
2873   \renewcommand*{\newacronymstyle}[1]{%
2874     \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
2875     available unless you restore the original acronym interface with
2876     \string\RestoreAcronyms}%
2877     \@glxtr@org@newacronymstyle{##1}%
2878   }%
2879 }
```

Switch acronyms to abbreviations:

```
2880 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```
2881 \newcommand*{\RestoreAcronyms}{%
2882   \SetGenericNewAcronym
2883   \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
2884   \renewcommand{\acronymfont}[1]{##1}%
2885   \let\setacronymstyle\@glxtr@org@setacronymstyle
2886   \let\newacronymstyle\@glxtr@org@newacronymstyle
```

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

```
2887 \renewcommand*\@gls@link@checkfirsthyper{%
2888   \ifglused{\glslabel}%
2889   {\let\glxtrifwasfirstuse\@secondoftwo}
2890   {\let\glxtrifwasfirstuse\@firstoftwo}%
2891   \@glxtr@org@checkfirsthyper
2892 }
2893 \glsssetcategoryattribute{acronym}{regular}{false}%
```

```

2894 \setacronymstyle{long-short}%
2895 }

```

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

```

2896 \renewcommand*{\glsacspace}[1]{%
2897   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
2898   \ifdim\dimen@<\glsacspacemax~\else\space\fi
2899 }

```

`\glsacspacemax` Value used in the above.

```

2900 \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`

```

2901 \newcommand*{\@glxtr@reg@glosslist}{}

```

Save the original definition of `\makeglossaries`:

```

2902 \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`

```

2903 \renewcommand*{\makeglossaries}[1] []{%
2904   \ifblank{#1}%
2905   {\@glxtr@org@makeglossaries}%
2906   {%
2907     \edef\@glxtr@reg@glosslist{#1}%
2908     \ifundef{\glswrite}{\newwrite\glswrite}{}%
2909     \protected@write\@auxout{}{\string\providecommand
2910       \string\@glxtr@reg@glosslist{#1}}
2911     \protected@write\@auxout{}{\string\providecommand
2912       \string\@glxtr@org@makeglossaries{#1}}
2913     \protected@write\@auxout{}{\string\@glxtr@reg@glosslist{#1}}
2914     \protected@write\@auxout{}{\string\@glxtr@org@makeglossaries{#1}}
2915     \protected@write\@auxout{}{\string\@glxtr@reg@glosslist{#1}}
2916     \write\@auxout{\string\providecommand\string\@glxtr@reg@glosslist{#1}}

```

Iterate through each supplied glossary type and activate it.

```

2917   \@for\@glo@type:=#1\do{%
2918     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
2919   }%

```

New glossaries must be created before `\makeglossaries`:

```
2920 \renewcommand*\newglossary[4] [] {%
2921 \PackageError{glossaries}{New glossaries
2922 must be created before \string\makeglossaries}{You need
2923 to move \string\makeglossaries\space after all your
2924 \string\newglossary\space commands}}%
```

Any subsequent instances of this command should have no effect

```
2925 \let\makeglossary\relax
2926 \let\makeglossary\relax
2927 \renewcommand\makeglossaries[1] [] {}%
```

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

```
2928 \@disable@onlypremakeg
```

Allow see key:

```
2929 \let\gls@checkseeallowed\relax
```

Adjust `\do@seeglossary`

```
2930 \let\glsxtr@org@doseeglossary\do@seeglossary
2931 \renewcommand*{\do@seeglossary}[2] {%
2932 \edef\gls@label{\glsdetoklabel{##1}}%
2933 \edef\gls@type{\csname glo@\gls@label @type\endcsname}%
2934 \expandafter\DTLifinlist\expandafter{\gls@type}{\glsxtr@reg@glosslist}%
2935 {\glsxtr@org@doseeglossary{##1}{##2}}%
2936 {%
2937 \protected@write\@auxout{}{%
2938 \string\gls@reference
2939 {\gls@type}{\gls@label}{\string\glsseeformat##2{}}}%
2940 }%
2941 }%
2942 }%
```

Adjust `\@do@@wrglossary`

```
2943 \let\glsxtr@@do@@wrglossary\@do@@wrglossary
2944 \def\@do@@wrglossary{%
2945 \edef\gls@type{\csname glo@\gls@label @type\endcsname}%
2946 \expandafter\DTLifinlist\expandafter{\gls@type}{\glsxtr@reg@glosslist}%
2947 {\glsxtr@@do@@wrglossary}%
2948 {\gls@noidxglossary}%
2949 }%
```

Suppress warning about no `\makeglossaries`

```
2950 \let\warn@nomakeglossaries\relax
2951 \def\warn@noprntglossary{%
2952 \GlossariesWarningNoLine{No \string\printglossary\space
2953 or \string\printglossaries\space
2954 found.^^J(Remove \string\makeglossaries\space if you don't want
2955 any glossaries.)^^JThis document will not have a glossary}%
2956 }%
```

Only warn for glossaries not listed.

```

2957 \renewcommand{\@gls@noref@warn}[1]{%
2958 \edef\@gls@type{##1}%
2959 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
2960 {%
2961 \GlossariesExtraWarning{Can't use
2962 \string\printnoidxglossary[type={\@gls@type}]
2963 when '\@gls@type' is listed in the optional argument of
2964 \string\makeglossaries}%
2965 }%
2966 {%
2967 \GlossariesWarning{Empty glossary for
2968 \string\printnoidxglossary[type={##1}].
2969 Rerun may be required (or you may have forgotten to use
2970 commands like \string\gls)}%
2971 }%
2972 }%

```

Adjust display number list to check for type:

```

2973 \renewcommand*\@glsdisplaynumberlist}[1]{%
2974 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2975 {\@glsxtr@idx@displaynumberlist{##1}}%
2976 {\@glsxtr@noidx@displaynumberlist{##1}}%
2977 }%

```

Adjust entry list:

```

2978 \renewcommand*\@glsentrynumberlist}[1]{%
2979 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2980 {\@glsxtr@idx@entrynumberlist{##1}}%
2981 {\@glsxtr@noidx@entrynumberlist{##1}}%
2982 }%

```

Adjust number list loop

```

2983 \renewcommand*\@glsnumberlistloop}[2]{%
2984 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2985 {%
2986 \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
2987 not available for glossary '##1'}{%
2988 }%
2989 {\@glsxtr@noidx@numberlistloop{##1}{##2}}%
2990 }%

```

Only sanitize sort for normal indexing glossaries.

```

2991 \renewcommand*\@glsprestandardsort}[3]{%
2992 \expandafter\DTLifinlist\expandafter{##2}{\@glsxtr@reg@glosslist}%
2993 {%
2994 \glsdosanitizesort
2995 }%
2996 {%
2997 \ifglssanitizesort
2998 \@gls@noidx@sanitizesort

```

```

2999     \else
3000     \@gls@noidx@nosanitizesort
3001     \fi
3002 }%
3003 }%

```

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

```

3004 \renewcommand*{\new@glossaryentry}[2]{%
3005   \PackageError{glossaries-extra}{Glossary entries must be defined
3006     in the preamble\MessageBreak when you use the optional argument
3007     of \string\makeglossaries}{Either move your definitions to the
3008     preamble or don't use the optional argument of
3009     \string\makeglossaries}%
3010 }%

```

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).

```

3011 \let\@glo@assign@sortkey\@glsxtr@mixed@assign@sortkey
3012 \renewcommand*{\@printgloss@setsort}{%

```

Need to extract just the type value.

```

3013   \expandafter\@glsxtr@gettype\expandafter,\@glsxtr@printglossopts,%
3014   type=\glsdefaulttype,\@end@glsxtr@gettype
3015   \def\@glo@sorttype{\@glo@default@sorttype}%
3016 }%

```

Check automake setting:

```

3017 \ifglsautomake
3018   \renewcommand*{\@gls@doautomake}{%
3019     \@for\@gls@type:=\@glsxtr@reg@glosslist\do{%
3020       \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3021     }%
3022   }%
3023 \fi
3024 }%
3025 }

```

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

`\@printglossary` This no longer simply saves `\@printglossary` with `\let` 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 `gls.tex` file for some settings, so the glossary might not have been defined.) (This command is also used for on-the-fly setting.)

```

3026 \newcommand{\@glsxtr@orgprintglossary}[2]{%
3027   \def\@glo@type{\glsdefaulttype}%

```

Add check here.

```

3028   \def\glossarytitle{%
3029     \ifcsdef{\@glo@type\@glo@type @title}%

```



```

3030     {\csuse{@glo@type@\glo@type @title}}%
3031     {\glossaryname}}%
3032 \def\glossarytoctitle{\glossarytitle}%
3033 \let\org@glossarytitle\glossarytitle
3034 \def\@glossarystyle{%
3035     \ifx\@glossary@default@style\relax
3036         \GlossariesWarning{No default glossary style provided \MessageBreak
3037             for the glossary '\@glo@type'. \MessageBreak
3038             Using deprecated fallback. \MessageBreak
3039             To fix this set the style with \MessageBreak
3040             \string\setglossarystyle\space or use the \MessageBreak
3041             style key=value option}%
3042     \fi
3043 }%
3044 \def\gls@dotocitle{\glssettocitle{\@glo@type}}%
3045 \let\@org@glossaryentrynumbers\glossaryentrynumbers
3046 \bgroup
3047     \@printgloss@setsort
3048     \setkeys{printgloss}{#1}%
3049     \ifx\glossarytitle\org@glossarytitle
3050     \else
3051         \cslet{@glo@type@\glo@type @title}{\glossarytitle}%
3052     \fi
3053     \let\currentglossary\@glo@type
3054     \let\org@glossaryentrynumbers\glossaryentrynumbers
3055     \let\glsnonextpages\@glsnonextpages
3056     \let\glsnextpages\@glsnextpages
3057     \let\nopostdesc\@nopostdesc
3058     \gls@dotocitle
3059     \@glossarystyle
3060     \let\gls@org@glossaryentryfield\glossentry
3061     \let\gls@org@glossarysubentryfield\subglossentry
3062     \renewcommand{\glossentry}[1]{%
3063         \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3064         \gls@org@glossaryentryfield{##1}%
3065     }%
3066     \renewcommand{\subglossentry}[2]{%
3067         \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3068         \gls@org@glossarysubentryfield{##1}{##2}%
3069     }%
3070     \@gls@preglossaryhook
3071     #2%
3072 \egroup
3073 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
3074 \global\let\warn@noprintglossary\relax
3075 }

```

\@printglossary Redefine.

```

3076 \renewcommand{\@printglossary}[2]{%

```

```

3077 \def\@glxtr@printglossopts{#1}%
3078 \@glxtr@orgprintglossary{#1}{#2}%
3079 }

```

Add a key that switches off the entry targets:

```

3080 \define@choicekey{printgloss}{target}[\val\nr]{true,false}[true]{%
3081   \ifcase\nr
3082     \let\@glstarget\glsdohypertarget
3083   \else
3084     \let\@glstarget\@secondoftwo
3085   \fi
3086 }

```

@makeglossaries For the benefit of makeglossaries

```

3087 \newcommand*{\@glxtr@makeglossaries}[1]{%

```

@glxtr@gettype Get just the type.

```

3088 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
3089   \def\@glo@type{#2}%
3090 }

```

@assign@sortkey Assign the sort key.

```

3091 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
3092   \edef\@glo@type{\@glo@type}%
3093   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
3094   {%
3095     \@glo@no@assign@sortkey{#1}%
3096   }%
3097   {%
3098     \@glo@assign@sortkey{#1}%
3099   }%
3100 }%

```

Display number list for the regular version:

splaynumberlist

```

3101 \let\@glxtr@idx@displaynumberlist\glstdisplaynumberlist

```

Display number list for the “noidx” version:

splaynumberlist

```

3102 \newcommand*{\@glxtr@noidx@displaynumberlist}[1]{%
3103   \letcs{\@gls@loclist}{glo@\glsetoklabel{#1}@loclist}%
3104   \ifdef\@gls@loclist
3105   {%
3106     \def\@gls@noidxloclist@sep{%
3107       \def\@gls@noidxloclist@sep{%
3108         \def\@gls@noidxloclist@sep{%
3109           \glsnumlistsep

```

```

3110     }%
3111     \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
3112     }%
3113     }%
3114     \def\@gls@noidxloclist@finalsep{}%
3115     \def\@gls@noidxloclist@prev{}%
3116     \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
3117     \@gls@noidxloclist@finalsep
3118     \@gls@noidxloclist@prev
3119 }%
3120 {%
3121   ??\glsdoifexists{#1}%
3122   {%
3123     \GlossariesWarning{Missing location list for ‘#1’. Either
3124       a rerun is required or you haven’t referenced the entry.}%
3125   }%
3126 }%
3127 }%
3128

```

And for the number list loop:

@numberlistloop

```

3129 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
3130   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3131   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
3132   \let\@gls@org@glsseeformat\glsseeformat
3133   \let\glsnoidxdisplayloc#2\relax
3134   \let\glsseeformat#3\relax
3135   \ifdef\@gls@loclist
3136   {%
3137     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3138   }%
3139   {%
3140     ??\glsdoifexists{#1}%
3141     {%
3142       \GlossariesWarning{Missing location list for ‘##1’. Either
3143         a rerun is required or you haven’t referenced the entry.}%
3144     }%
3145   }%
3146   \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3147   \let\glsseeformat\@gls@org@glsseeformat
3148 }%

```

Same for entry number list.

entrynumberlist

```

3149 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
3150   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3151   \ifdef\@gls@loclist

```

```

3152 {%
3153   \glsnoidxloclist{\@gls@loclist}%
3154 }%
3155 {%
3156   ??\glsdoifexists{#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 }%

```

entrynumberlist

```

3163 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

@noidx@glossary

```

3164 \renewcommand*{\@print@noidx@glossary}{%
3165   \ifcsdef{\@glsref@{\@glo@type}}%
3166   {%
3167     \ifcsdef{\@glo@sortmacro@{\@glo@sorttype}}%
3168     {%
3169       \csuse{\@glo@sortmacro@{\@glo@sorttype}}{\@glo@type}%
3170     }%
3171     {%
3172       \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
3173     }%
3174     \glossarysection[\glossarytoctitle]{\glossarytitle}%
3175     \glossarypreamble

```

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

```

3176   \def\@gls@currentlettergroup{}%
3177   \begin{theglossary}%
3178   \glossaryheader
3179   \glsresetentrylist
3180   \forlistcsloop{\@gls@noidx@do}{\@glsref@{\@glo@type}}%
3181   \end{theglossary}%
3182   \glossarypostamble
3183 }%
3184 {%

```

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

```

3185   \glsxtrifemptyglossary{\@glo@type}%
3186   {}%
3187   {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
3188   \@gls@noref@warn{\@glo@type}%
3189 }%
3190 }

```

noidxdisplayloc Patch to check for range formations.

```
3191 \renewcommand*{\glsnoidxdisplayloc}[4]{%
3192   \setentrycounter[#1]{#2}%
3193   \@glsxtr@display@loc#3\empty\end@glsxtr@display@loc{#4}%
3194 }
```

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

```
3195 \def\@glsxtr@display@loc#1#2\end@glsxtr@display@loc#3{%
3196   \ifx#1(\relax
3197     \glsxtrdisplaystartloc{#2}{#3}%
3198   \else
3199     \ifx#1)\relax
3200       \glsxtrdisplayendloc{#2}{#3}%
3201     \else
3202       \glsxtrdisplaysingleloc{#1#2}{#3}%
3203     \fi
3204   \fi
3205 }
```

isplaysingleloc Single location.

```
3206 \newcommand*{\glsxtrdisplaysingleloc}[2]{%
3207   \csuse{#1}{#2}%
3208 }
```

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

displaystartloc Start of a location range.

```
3209 \newcommand*{\glsxtrdisplaystartloc}[2]{%
3210   \edef\glsxtrlocrangefmt{#1}%
3211   \ifx\glsxtrlocrangefmt\empty
3212     \def\glsxtrlocrangefmt{\glsnumberformat}%
3213   \fi
3214   \expandafter\glsxtrdisplaysingleloc
3215   \expandafter{\glsxtrlocrangefmt}{#2}%
3216 }
```

trdisplayendloc End of a location range.

```
3217 \newcommand*{\glsxtrdisplayendloc}[2]{%
3218   \ifdefstring{\glsxtrlocrangefmt}{#1}{}%
3219   {\GlossariesExtraWarning{Mismatched end location range
3220     (start=\glsxtrlocrangefmt, end=#1)}}%
3221   }%
3222   \glsxtrdisplayendloohook{#1}{#2}%
3223   \expandafter\glsxtrdisplaysingleloc
3224   \expandafter{\glsxtrlocrangefmt}{#2}%
3225   \def\glsxtrlocrangefmt{}%
3226 }
```

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

```
3227 \newcommand*{\glxtrdisplayendlochook}[2]{}
```

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

```
3228 \newcommand*{\glxsxtrlocrangefmt}{}%
```

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

@print@glossary

```
3229 \renewcommand{\@print@glossary}{%
3230   \makeatletter
3231   \@input{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
3232   \IfFileExists{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
3233   {}%
3234   {\glxtrNoGlossaryWarning{\@glo@type}}%
3235   \ifglxsindy
3236     \ifcsundef{@xdy@\@glo@type @language}%
3237     {%
3238       \edef\@do@auxoutstuff{%
3239         \noexpand\AtEndDocument{%
3240           \noexpand\immediate\noexpand\write\@auxout{%
3241             \string\providecommand\string\@xdylanguage[2]{}}%
3242           \noexpand\immediate\noexpand\write\@auxout{%
3243             \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
3244           }%
3245         }%
3246       }%
3247     }%
3248     \edef\@do@auxoutstuff{%
3249       \noexpand\AtEndDocument{%
3250         \noexpand\immediate\noexpand\write\@auxout{%
3251           \string\providecommand\string\@xdylanguage[2]{}}%
3252         \noexpand\immediate\noexpand\write\@auxout{%
3253           \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
3254             @language\endcsname}}%
3255         }%
3256       }%
3257     }%
3258     \@do@auxoutstuff
3259     \edef\@do@auxoutstuff{%
3260       \noexpand\AtEndDocument{%
3261         \noexpand\immediate\noexpand\write\@auxout{%
3262           \string\providecommand\string\@gls@codepage[2]{}}%
3263         \noexpand\immediate\noexpand\write\@auxout{%
3264           \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
3265         }%
3266       }%
3267     \@do@auxoutstuff
```

```

3268 \fi
3269 \renewcommand*{\@warn@nomakeglossaries}{%
3270   \GlossariesWarningNoLine{\string\makeglossaries\space
3271   hasn't been used,^^Jthe glossaries will not be updated}%
3272 }%
3273 }

```

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

`\GlsWarningHead` Header message.

```

3274 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
3275   This document is incomplete. The external file associated with
3276   the glossary '#1' (which should be called \texttt{#2})
3277   hasn't been created.%
3278 }

```

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

```

3279 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
3280   This has probably happened because there are no entries defined
3281   in this glossary.%
3282 }

```

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

```

3283 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
3284   If you don't want this glossary,
3285   add \texttt{nomain} to your package option list when you load
3286   \texttt{glossaries-extra.sty}. For example:%
3287 }

```

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

```

3288 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
3289   Did you forget to use \texttt{type=#1} when you defined your
3290   entries? If you tried to load entries into this glossary with
3291   \texttt{\string\loadglsentries} did you remember to use
3292   \texttt{[#1]} as the optional argument? If you did, check that
3293   the definitions in the file you loaded all had the type set
3294   to \texttt{\string\glsdefaulttype}.%
3295 }

```

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

```

3296 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
3297   Check the contents of the file \texttt{#1}. If
3298   it's empty, that means you haven't indexed any of your entries in this
3299   glossary (using commands like \texttt{\string\gls} or
3300   \texttt{\string\glsadd}) so this list can't be generated.
3301   If the file isn't empty, the document build process hasn't been
3302   completed.%
3303 }

```

WarningAutoMake Message when automake option has been used.

```
3304 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
3305   You may need to rerun \LaTeX. If you already have, it may be that
3306   \TeX's shell escape doesn't allow you to run
3307   \ifglxindy xindy\else makeindex\fi. Check the
3308   transcript file \texttt{\jobname.log}. If the shell escape is
3309   disabled, try one of the following:
3310
3311   \begin{itemize}
3312     \item Run the external (Lua) application:
3313
3314       \texttt{makeglossaries-lite.lua \string\jobname\string}
3315
3316     \item Run the external (Perl) application:
3317
3318       \texttt{makeglossaries \string\jobname\string}
3319   \end{itemize}
3320
3321   Then rerun \LaTeX\ on this document.
3322   \GlossariesExtraWarning{Rerun required to build the
3323   glossary '#1' or check TeX's shell escape allows
3324   you to run \ifglxindy xindy\else makeindex\fi}%
3325 }
```

WarningMismatch Mismatching \makenoidxglossaries.

```
3326 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
3327   You need to either replace \texttt{\string\makenoidxglossaries}
3328   with \texttt{\string\makeglossaries} or replace
3329   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
3330   \texttt{\string\printnoidxglossary}
3331   (or \texttt{\string\printnoidxglossaries}) and then rebuild
3332   this document.%
3333 }
```

WarningBuildInfo Build advice.

```
3334 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
3335   Try one of the following:
3336   \begin{itemize}
3337     \item Add \texttt{automake} to your package option list when you load
3338       \texttt{glossaries-extra.sty}. For example:
3339
3340       \texttt{\string\usepackage[automake]%
3341       \glsopenbrace glossaries-extra\glsclosebrace}
3342
3343     \item Run the external (Lua) application:
3344
3345       \texttt{makeglossaries-lite.lua \string\jobname\string}
3346
```



```

3347 \item Run the external (Perl) application:
3348
3349 \texttt{makeglossaries \string\jobname\string}
3350 \end{itemize}
3351
3352 Then rerun \LaTeX\ on this document.%
3353 }

```

oGlsWarningTail Final paragraph.

```

3354 \newcommand{\GlsXtrNoGlsWarningTail}{%
3355 This message will be removed once the problem has been fixed.%
3356 }

```

GlsWarningNoOut No out file created. Build advice.

```

3357 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
3358 The file \texttt{#1} doesn't exist. This most likely means you haven't used
3359 \texttt{\string\makeglossaries} or you have used
3360 \texttt{\string\nofiles}. If this is just a draft version of the
3361 document, you can suppress this message using the
3362 \texttt{\nomissingglstext} package option.%
3363 }

```

glossarywarning

```

3364 \newcommand*{@@glsxtr@defaultnoglossarywarning}[1]{%
3365 \glossarysection[\glossarytoctitle]{\glossarytitle}
3366 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glo@type @in\endcsname}
3367 \par
3368 \glsxtrifemptyglossary{#1}%
3369 {%
3370 \GlsXtrNoGlsWarningEmptyStart\space
3371 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
3372 \medskip
3373 \noindent\texttt{\string\usepackage[nomain\ifglsacronym ,acronym\fi]%
3374 \glsopenbrace glossaries-extra\glsclosebrace}
3375 \medskip
3376 }%
3377 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
3378 }%
3379 {%
3380 \IfFileExists{\jobname.\csname @glo@type @out\endcsname}
3381 {%
3382 \GlsXtrNoGlsWarningCheckFile
3383 {\jobname.\csname @glo@type @out\endcsname}
3384
3385 \ifglsautomake
3386
3387 \GlsXtrNoGlsWarningAutoMake{#1}
3388
3389 \else

```

```

3390
3391     \ifthenelse{\equal{#1}{main}}{%
3392     {%
3393         \GlsXtrNoGlsWarningEmptyMain\par
3394         \medskip
3395         \noindent\texttt{\string\usepackage[nomain]%
3396             \glsopenbrace glossaries-extra\glsclosebrace}
3397         \medskip
3398     }%
3399     }%
3400
3401     \ifdefequal\makeglossaries\@no@makeglossaries
3402     {%
3403         \GlsXtrNoGlsWarningMismatch
3404     }%
3405     {%
3406         \GlsXtrNoGlsWarningBuildInfo
3407     }%
3408     \fi
3409 }%
3410 {%
3411     \GlsXtrNoGlsWarningNoOut
3412     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
3413 }%
3414 }%
3415 \par
3416 \GlsXtrNoGlsWarningTail
3417 }

```

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

glstxrresourcefile 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.

```

3418 \newcommand*{\glstxrresourcefile}[2] [] {%
3419     \protected@write\@auxout{}{\string\glstxr@resource{#1}{#2}}%
3420     \glstxr@writefields
3421     \let\@glstxr@org@see@noindex\@gls@see@noindex
3422     \let\@gls@see@noindex\relax
3423     \IfFileExists{#2.glstex}%
3424     {%

```

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

```

3425     \edef\@bibgls@restreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
3426     \makeatletter
3427     \@input{#2.glstex}%
3428     \@bibgls@restreat
3429 }%
3430 {%
3431     \GlossariesExtraWarning{No file '#2.glstex'}%

```

```

3432 }%
3433 \let\@gls@see@noindex\@glsxtr@org@see@noindex
3434 }
3435 \@onlypreamble\glsxtrresourcefile

trresourcecount
3436 \newcount\glsxtrresourcecount

trLoadResources Short cut that uses \glsxtrresourcefile with \jobname as the mandatory argument.
3437 \newcommand*\GlsXtrLoadResources}[1][{}]{%
3438   \ifnum\glsxtrresourcecount=0\relax
3439     \glsxtrresourcefile[#1]{\jobname}%
3440   \else
3441     \glsxtrresourcefile[#1]{\jobname-\the\glsxtrresourcecount}%
3442   \fi
3443   \advance\glsxtrresourcecount by 1\relax
3444 }

glsxtr@resource
3445 \newcommand*\glsxtr@resource}[2]{}

\glsxtr@fields
3446 \newcommand*\glsxtr@fields}[1]{}

xtr@texencoding
3447 \newcommand*\glsxtr@texencoding}[1]{}

\glsxtr@langtag
3448 \newcommand*\glsxtr@langtag}[1]{}

@pluralsuffixes
3449 \newcommand*\glsxtr@pluralsuffixes}[4]{}

tr@shortcutsval
3450 \newcommand*\glsxtr@shortcutsval}[1]{}

sxtr@linkprefix
3451 \newcommand*\glsxtr@linkprefix}[1]{}

xtr@writefields This information only needs to be written once, so disable it after it's been used.
3452 \newcommand*\glsxtr@writefields){%
3453   \protected@write\@auxout{}\string\glsxtr@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 \glsxtrresourcefile.
3454   \ifdef\CurrentTrackedLanguageTag
3455     {%

```

```

3456 \protected@write\@auxout{}\{%
3457 \string\glxtr@langtag{\CurrentTrackedLanguageTag}}%
3458 }%
3459 {%
3460 \protected@write\@auxout{}\{\string\glxtr@pluralsuffixes
3461 {\glsppluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
3462 {\glxtrabbrvpluralsuffix}}%
3463 \ifdef\inputencodingname
3464 {%
3465 \protected@write\@auxout{}\{\string\glxtr@texencoding{\inputencodingname}}%
3466 }%
3467 {%

```

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.)

```

3468 \ifpackageloaded{fontspec}%
3469 {\protected@write\@auxout{}\{\string\glxtr@texencoding{utf8}}}%
3470 }%
3471 }%
3472 \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.

```

3473 \AtBeginDocument
3474 {\protected@write\@auxout{}\{\string\glxtr@linkprefix{\glolinkprefix}}}%
3475 \let\glxtr@writefields\relax
3476 }

```

Allow locations specific to a particular counter to be recorded.

`\glxtr@record`

```

3477 \newcommand*\glxtr@record[5]{%

```

`\@counterrecord` Aux file command.

```

3478 \newcommand*\glxtr@counterrecord[3]{%
3479 \glxtrfieldlistgadd{#1}{record.#2}{#3}%
3480 }

```

`\@counterrecordhook` Hook used by `\glxtr@dorecord`.

```

3481 \newcommand*\@glxtr@counterrecordhook{}

```

`\glxtrRecordCounter` Activate recording for a particular counter (identified in the argument).

```

3482 \newcommand*\GlsXtrRecordCounter[1]{%
3483 \@@glxtr@recordcounter{#1}%
3484 }
3485 \onlypreamble\GlsXtrRecordCounter

```

`\docounterrecord`

```

3486 \newcommand*\@glxtr@docounterrecord[1]{%

```

```

3487 \protected@write\@auxout{}\string\glxtr@counterrecord
3488 {\@gls@label}{#1}{\csuse{the#1}}}%
3489 }

\printunsrtglossary Similar to \printnoidxglossary but it displays all entries defined for the given glossary
without sorting.
3490 \newcommand*\printunsrtglossary{%
3491 \ifstar\s@printunsrtglossary\printunsrtglossary
3492 }

\printunsrtglossary Unstarred version.
3493 \newcommand*\@printunsrtglossary[1][\]{%
3494 \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
3495 }

\printunsrtglossary Starred version.
3496 \newcommand*\s@printunsrtglossary[2][\]{%
3497 \begin{group}
3498 #2%
3499 \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
3500 \end{group}
3501 }

\printunsrtglossaries Similar to \printnoidxglossaries but it displays all entries defined for the given glossary
without sorting.
3502 \newcommand*\printunsrtglossaries{%
3503 \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
3504 }

\@unsrt@glossary
3505 \newcommand*\@print@unsrt@glossary{%
3506 \glossarysection[\glossarytoctitle]{\glossarytitle}%
3507 \glossarypreamble
check for empty list
3508 \glxtrifemptyglossary{\@glo@type}%
3509 {%
3510 \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
3511 }%
3512 {%
3513 \key@ifundefined{glossentry}{group}%
3514 {\let\@gls@getgrouptitle\@glxtr@noidx@getgrouptitle}%
3515 {\let\@gls@getgrouptitle\@glxtr@unsrt@getgrouptitle}%
3516 \begin{theglossary}%
3517 \glossaryheader
3518 \glsresetentrylist
3519 \def\@gls@currentlettergroup{}%
3520 \expandafter\@for\expandafter\glscurrententrylabel\expandafter
3521 : \expandafter=\csname glist@\@glo@type\endcsname\do{%

```

```

3522     \ifdefempty{\glscurrententrylabel}
3523     {}%
3524     {\printunsrtglossaryhandler\glscurrententrylabel}%
3525     }%
3526     \end{theglossary}%
3527     }%
3528     \glossarypostamble
3529 }

```

glossaryhandler

```

3530 \newcommand{\printunsrtglossaryhandler}[1]{%
3531   \glxtrunsrtdo{#1}%
3532 }

```

srtglossaryunit

```

3533 \newcommand{\print@op@unsrtglossaryunit}[2][]{%
3534   \s@printunsrtglossary[type=\gldefaulttype,#1]{%
3535     \printunsrtglossaryunitsetup{#2}%
3536   }%
3537 }

```

ossaryunitsetup

```

3538 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
3539   \renewcommand{\printunsrtglossaryhandler}[1]{%
3540     \glxtrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
3541     {\glxtrunsrtdo{##1}}%
3542     {}%
3543   }%
3544   \ifcsundef{theH#1}%
3545   {%
3546     \renewcommand*{\glolinkprefix}{record.#1.\csuse{the#1}.}%
3547   }%
3548   {%
3549     \renewcommand*{\glolinkprefix}{record.#1.\csuse{theH#1}.}%
3550   }%
3551   \renewcommand*{\glossarysection}[2][]{}%
3552   \appto\glossarypostamble{\glspare\medskip\glspare}%
3553 }

```

srtglossaryunit

```

3554 \newcommand{\print@noop@unsrtglossaryunit}[2][]{%
3555   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
3556     requires the record=only or record=alsoindex package option}{}%
3557 }

```

t@getgrouptitle

```

3558 \newcommand*{\@glxtr@unsrt@getgrouptitle}[2]{%
3559   \def#2{#1}%
3560 }

```

```

\glxtrunsrtdo Provide a user-level call to \@glxtr@noidx@do to make it easier to define a new handler.
3561 \newcommand{\glxtrunsrtdo}{\@glxtr@noidx@do}

glxtr@noidx@do Minor modification of \@gls@noidx@do to check for location field if present.
3562 \newcommand{\@glxtr@noidx@do}[1]{%
3563   \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3564   \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@location}%
3565   \ifglshasparent{#1}%
3566   {%
3567     \gls@level=\csuse{glo@\glsdetoklabel{#1}@level}\relax
3568     \ifdefvoid{\@gls@location}%
3569     {%
3570       \ifdefvoid{\@gls@loclist}%
3571       {%
3572         \subglossentry{\gls@level}{#1}{}%
3573       }%
3574       {%
3575         \subglossentry{\gls@level}{#1}%
3576         {%
3577           \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
3578         }%
3579       }%
3580     }%
3581     {%
3582       \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
3583     }%
3584   }%
3585   {%
3586     \letcs{\@gls@sort}{glo@\glsdetoklabel{#1}@sort}%
3587     \key@ifundefined{glossentry}{group}%
3588     {%
3589       \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
3590     }%
3591     {%
3592       \protected@xdef\@glo@thislettergrp{%
3593         \csname glo@\glsdetoklabel{#1}@group\endcsname}%
3594       }%
3595       \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
3596       {}%
3597       {%
3598         \ifdefempty{\@gls@currentlettergroup}{}\@gls@groupskip%
3599         \gls@groupheading{\@glo@thislettergrp}%
3600       }%
3601       \let\@gls@currentlettergroup\@glo@thislettergrp
3602       \ifdefvoid{\@gls@location}%
3603       {%
3604         \ifdefvoid{\@gls@loclist}
3605         {%
3606           \glossentry{#1}{}%

```

```

3607     }%
3608     {%
3609         \glossentry{#1}%
3610         {%
3611             \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
3612             }%
3613         }%
3614     }%
3615     {%
3616         \glossentry{#1}%
3617         {%
3618             \glossaryentrynumbers{\@gls@location}%
3619             }%
3620         }%
3621     }%
3622 }

```

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.

```

3623 \ifpackageloaded{glossaries-accsupp}
3624 {

```

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

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

```

3625 \newcommand*{\glsaccessname}[1]{%
3626     \glsnameaccessdisplay
3627     {%
3628         \glsentryname{#1}%
3629         }%
3630     {#1}%
3631 }

```

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

```

3632 \newcommand*{\Glsaccessname}[1]{%
3633     \glsnameaccessdisplay
3634     {%
3635         \Glsentryname{#1}%
3636         }%
3637     {#1}%
3638 }

```

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


```

3639 \newcommand*{\GLSaccessname}[1]{%
3640   \glsnameaccessdisplay
3641   {%
3642     \mfirstucMakeUppercase{\glsentryname{#1}}%
3643   }%
3644   {#1}%
3645 }

```

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

```

3646 \newcommand*{\glsaccesstext}[1]{%
3647   \glstextaccessdisplay
3648   {%
3649     \glsentrytext{#1}%
3650   }%
3651   {#1}%
3652 }

```

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

```

3653 \newcommand*{\Glsaccesstext}[1]{%
3654   \glstextaccessdisplay
3655   {%
3656     \Glsentrytext{#1}%
3657   }%
3658   {#1}%
3659 }

```

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

```

3660 \newcommand*{\GLSaccesstext}[1]{%
3661   \glstextaccessdisplay
3662   {%
3663     \mfirstucMakeUppercase{\glsentrytext{#1}}%
3664   }%
3665   {#1}%
3666 }

```

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

```

3667 \newcommand*{\glsaccessplural}[1]{%
3668   \glspluralaccessdisplay
3669   {%
3670     \glsentryplural{#1}%
3671   }%
3672   {#1}%
3673 }

```

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

```

3674 \newcommand*{\Glsaccessplural}[1]{%

```

```

3675 \glspluralaccessdisplay
3676 {%
3677 \Glsentryplural{#1}%
3678 }%
3679 {#1}%
3680 }

```

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

```

3681 \newcommand*{\GLSaccessplural}[1]{%
3682 \glspluralaccessdisplay
3683 {%
3684 \mfirstucMakeUppercase{\glsentryplural{#1}}%
3685 }%
3686 {#1}%
3687 }

```

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

```

3688 \newcommand*{\glsaccessfirst}[1]{%
3689 \glsfirstaccessdisplay
3690 {%
3691 \glsentryfirst{#1}%
3692 }%
3693 {#1}%
3694 }

```

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

```

3695 \newcommand*{\Glsaccessfirst}[1]{%
3696 \glsfirstaccessdisplay
3697 {%
3698 \Glsentryfirst{#1}%
3699 }%
3700 {#1}%
3701 }

```

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

```

3702 \newcommand*{\GLSaccessfirst}[1]{%
3703 \glsfirstaccessdisplay
3704 {%
3705 \mfirstucMakeUppercase{\glsentryfirst{#1}}%
3706 }%
3707 {#1}%
3708 }

```

cessfirstplural Display the firstplural value (no link and no check for existence).

```

3709 \newcommand*{\glsaccessfirstplural}[1]{%
3710 \glsfirstpluralaccessdisplay
3711 {%

```

```

3712     \glentryfirstplural{#1}%
3713   }%
3714   {#1}%
3715 }

```

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

```

3716 \newcommand*{\Glsaccessfirstplural}[1]{%
3717   \glfirstpluralaccessdisplay
3718   {%
3719     \glentryfirstplural{#1}%
3720   }%
3721   {#1}%
3722 }

```

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

```

3723 \newcommand*{\GLSaccessfirstplural}[1]{%
3724   \glfirstpluralaccessdisplay
3725   {%
3726     \mfirstucMakeUppercase{\glentryfirstplural{#1}}%
3727   }%
3728   {#1}%
3729 }

```

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

```

3730 \newcommand*{\glaccesssymbol}[1]{%
3731   \glssymbolaccessdisplay
3732   {%
3733     \glentrysymbol{#1}%
3734   }%
3735   {#1}%
3736 }

```

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

```

3737 \newcommand*{\GLSaccesssymbol}[1]{%
3738   \glssymbolaccessdisplay
3739   {%
3740     \glentrysymbol{#1}%
3741   }%
3742   {#1}%
3743 }

```

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

```

3744 \newcommand*{\GLSaccesssymbol}[1]{%
3745   \glssymbolaccessdisplay
3746   {%
3747     \mfirstucMakeUppercase{\glentrysymbol{#1}}%

```

```

3748     }%
3749     {#1}%
3750 }

```

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

```

3751 \newcommand*{\glsaccesssymbolplural}[1]{%
3752   \glssymbolpluralaccessdisplay
3753   {%
3754     \glsentrysymbolplural{#1}%
3755   }%
3756   {#1}%
3757 }

```

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

```

3758 \newcommand*{\Glsaccesssymbolplural}[1]{%
3759   \glssymbolpluralaccessdisplay
3760   {%
3761     \Glsentrysymbolplural{#1}%
3762   }%
3763   {#1}%
3764 }

```

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

```

3765 \newcommand*{\GLSaccesssymbolplural}[1]{%
3766   \glssymbolpluralaccessdisplay
3767   {%
3768     \mfirstucMakeUppercase{\glsentrysymbolplural{#1}}%
3769   }%
3770   {#1}%
3771 }

```

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

```

3772 \newcommand*{\glsaccessdesc}[1]{%
3773   \glsdescriptionaccessdisplay
3774   {%
3775     \glsentrydesc{#1}%
3776   }%
3777   {#1}%
3778 }

```

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

```

3779 \newcommand*{\Glsaccessdesc}[1]{%
3780   \glsdescriptionaccessdisplay
3781   {%
3782     \Glsentrydesc{#1}%
3783   }%

```

```

3784     {#1}%
3785 }

```

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

```

3786 \newcommand*{\GLSaccessdesc}[1]{%
3787   \glsdescriptionaccessdisplay
3788   {%
3789     \mfirstucMakeUppercase{\glsentrydesc{#1}}%
3790   }%
3791   {#1}%
3792 }

```

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

```

3793 \newcommand*{\glsaccessdescplural}[1]{%
3794   \glsdescriptionpluralaccessdisplay
3795   {%
3796     \glsentrydescplural{#1}%
3797   }%
3798   {#1}%
3799 }

```

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

```

3800 \newcommand*{\Glsaccessdescplural}[1]{%
3801   \glsdescriptionpluralaccessdisplay
3802   {%
3803     \Glsentrydescplural{#1}%
3804   }%
3805   {#1}%
3806 }

```

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

```

3807 \newcommand*{\GLSaccessdescplural}[1]{%
3808   \glsdescriptionpluralaccessdisplay
3809   {%
3810     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
3811   }%
3812   {#1}%
3813 }

```

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

```

3814 \newcommand*{\glsaccessshort}[1]{%
3815   \glsshortaccessdisplay
3816   {%
3817     \glsentryshort{#1}%
3818   }%
3819   {#1}%
3820 }

```

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

```

3821 \newcommand*\Glsaccesssshort}[1]{%
3822   \glsshortaccessdisplay
3823   {%
3824     \Glsentryshort{#1}%
3825   }%
3826   {#1}%
3827 }
```

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

```

3828 \newcommand*\GLSaccesssshort}[1]{%
3829   \glsshortaccessdisplay
3830   {%
3831     \mfirstucMakeUppercase{\Glsentryshort{#1}}%
3832   }%
3833   {#1}%
3834 }
```

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

```

3835 \newcommand*\lsaccesssshortpl}[1]{%
3836   \glsshortpluralaccessdisplay
3837   {%
3838     \Glsentryshortpl{#1}%
3839   }%
3840   {#1}%
3841 }
```

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

```

3842 \newcommand*\lsaccesssshortpl}[1]{%
3843   \glsshortpluralaccessdisplay
3844   {%
3845     \Glsentryshortpl{#1}%
3846   }%
3847   {#1}%
3848 }
```

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

```

3849 \newcommand*\LSaccesssshortpl}[1]{%
3850   \glsshortpluralaccessdisplay
3851   {%
3852     \mfirstucMakeUppercase{\Glsentryshortpl{#1}}%
3853   }%
3854   {#1}%
3855 }
```

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

```

3856 \newcommand*\glsaccesslong}[1]{%
3857   \glslongaccessdisplay{\glsentrylong{#1}}{#1}%
3858 }

```

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

```

3859
3860 \newcommand*\Glsaccesslong}[1]{%
3861   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
3862 }

```

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

```

3863 \newcommand*\GLSaccesslong}[1]{%
3864   \glslongaccessdisplay
3865   {%
3866     \mfirstucMakeUppercase{\glsentrylong{#1}}%
3867   }%
3868   {#1}%
3869 }

```

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

```

3870 \newcommand*\glsaccesslongpl}[1]{%
3871   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
3872 }

```

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

```

3873
3874 \newcommand*\Glsaccesslongpl}[1]{%
3875   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
3876 }

```

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

```

3877 \newcommand*\GLSaccesslongpl}[1]{%
3878   \glslongpluralaccessdisplay
3879   {%
3880     \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
3881   }%
3882   {#1}%
3883 }

```

End of if part

```

3884 }
3885 {

```

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

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

```

3886 \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.
3887 `\newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}`

`\GLSaccessname` Display the name value (no link and no check for existence). converted to upper case.
3888 `\newcommand*{\GLSaccessname}[1]{%`
3889 `\protect\mfirstucMakeUppercase{\glentryname{#1}}}`

`\glsaccesstext` Display the text value (no link and no check for existence).
3890 `\newcommand*{\glsaccesstext}[1]{\glentrytext{#1}}`

`\Glsaccesstext` Display the text value (no link and no check for existence) with the first letter converted to upper case.
3891 `\newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}`

`\GLSaccesstext` Display the text value (no link and no check for existence). converted to upper case.
3892 `\newcommand*{\GLSaccesstext}[1]{%`
3893 `\protect\mfirstucMakeUppercase{\glentrytext{#1}}}`

`glsaccessplural` Display the plural value (no link and no check for existence).
3894 `\newcommand*{\glsaccessplural}[1]{\glentryplural{#1}}`

`GLsaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.
3895 `\newcommand*{\GLsaccessplural}[1]{\GLsentryplural{#1}}`

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.
3896 `\newcommand*{\GLSaccessplural}[1]{%`
3897 `\protect\mfirstucMakeUppercase{\glentryplural{#1}}}`

`\glsaccessfirst` Display the first value (no link and no check for existence).
3898 `\newcommand*{\glsaccessfirst}[1]{\glentryfirst{#1}}`

`\GLsaccessfirst` Display the first value (no link and no check for existence) with the first letter converted to upper case.
3899 `\newcommand*{\GLsaccessfirst}[1]{\GLsentryfirst{#1}}`

`\GLSaccessfirst` Display the first value (no link and no check for existence). converted to upper case.
3900 `\newcommand*{\GLSaccessfirst}[1]{%`
3901 `\protect\mfirstucMakeUppercase{\glentryfirst{#1}}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence).
3902 `\newcommand*{\glsaccessfirstplural}[1]{\glentryfirstplural{#1}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.
3903 `\newcommand*{\GLsaccessfirstplural}[1]{\GLsentryfirstplural{#1}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.
3904 `\newcommand*{\GLSaccessfirstplural}[1]{%`
3905 `\protect\mfirstucMakeUppercase{\glentryfirstplural{#1}}}`

`glsaccesssymbol` Display the symbol value (no link and no check for existence).
3906 `\newcommand*{\glsaccesssymbol}[1]{\glentrysymbol{#1}}`

`GLsaccesssymbol` Display the symbol value (no link and no check for existence) with the first letter converted to upper case.
3907 `\newcommand*{\GLsaccesssymbol}[1]{\GLentrysymbol{#1}}`

`GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.
3908 `\newcommand*{\GLSaccesssymbol}[1]{%`
3909 `\protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}`

`esssymbolplural` Display the symbolplural value (no link and no check for existence).
3910 `\newcommand*{\glsaccesssymbolplural}[1]{\glentrysymbolplural{#1}}`

`esssymbolplural` Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.
3911 `\newcommand*{\GLsaccesssymbolplural}[1]{\GLentrysymbolplural{#1}}`

`esssymbolplural` Display the symbolplural value (no link and no check for existence). converted to upper case.
3912 `\newcommand*{\GLSaccesssymbolplural}[1]{%`
3913 `\protect\mfirstucMakeUppercase{\glentrysymbolplural{#1}}}`

`\glsaccessdesc` Display the desc value (no link and no check for existence).
3914 `\newcommand*{\glsaccessdesc}[1]{\glentrydesc{#1}}`

`\GLsaccessdesc` Display the desc value (no link and no check for existence) with the first letter converted to upper case.
3915 `\newcommand*{\GLsaccessdesc}[1]{\GLentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence). converted to upper case.
3916 `\newcommand*{\GLSaccessdesc}[1]{%`
3917 `\protect\mfirstucMakeUppercase{\glentrydesc{#1}}}`

`ccessdescplural` Display the descplural value (no link and no check for existence).
3918 `\newcommand*{\glsaccessdescplural}[1]{\glentrydescplural{#1}}`

`ccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.
3919 `\newcommand*{\GLsaccessdescplural}[1]{\GLentrydescplural{#1}}`

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

```

3920 \newcommand*{\GLSaccessdescplural}[1]{%
3921 \protect\mfirstucMakeUppercase{\glstrydescplural{#1}}}
```

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

```

3922 \newcommand*{\glsaccessshort}[1]{\glstryshort{#1}}
```

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

```

3923 \newcommand*{\Glsaccessshort}[1]{\Glsstryshort{#1}}
```

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

```

3924 \newcommand*{\GLSaccessshort}[1]{%
3925 \protect\mfirstucMakeUppercase{\glstryshort{#1}}}
```

lsaccessshortpl Display the short plural form (no link and no check for existence).

```

3926 \newcommand*{\glsaccessshortpl}[1]{\glstryshortpl{#1}}
```

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

```

3927 \newcommand*{\Glsaccessshortpl}[1]{\Glsstryshortpl{#1}}
```

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

```

3928 \newcommand*{\GLSaccessshortpl}[1]{%
3929 \protect\mfirstucMakeUppercase{\glstryshortpl{#1}}}
```

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

```

3930 \newcommand*{\glsaccesslong}[1]{\glstrylong{#1}}
```

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

```

3931 \newcommand*{\Glsaccesslong}[1]{\Glsstrylong{#1}}
```

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

```

3932 \newcommand*{\GLSaccesslong}[1]{%
3933 \protect\mfirstucMakeUppercase{\glstrylong{#1}}}
```

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

```

3934 \newcommand*{\glsaccesslongpl}[1]{\glstrylongpl{#1}}
```

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

```

3935 \newcommand*{\Glsaccesslongpl}[1]{\Glsstrylongpl{#1}}
```

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

```

3936 \newcommand*{\GLSaccesslongpl}[1]{%
3937 \protect\mfirstucMakeUppercase{\glstrylongpl{#1}}}
```

End of else part

```

3938 }
```

1.5 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.

```
3939 \glsaddstoragekey{category}{general}{\glscategory}
```

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.

```
3940 \newcommand{\glsifcategory}[4]{%
3941   \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
3942 }
```

Categories can have attributes.

`categoryattribute` `\glssetcategoryattribute{<category>}{<attribute-label>}{<value>}`

Set (or override if already set) an attribute for the given category.

```
3943 \newcommand*\glssetcategoryattribute[3]{%
3944   \csdef{@glxtr@categoryattr@@#1@#2}{#3}%
3945 }
```

`categoryattribute` `\glsgetcategoryattribute{<category>}{<attribute-label>}`

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
3946 \newcommand*\glsgetcategoryattribute[2]{%
3947   \csuse{@glxtr@categoryattr@@#1@#2}%
3948 }
```

`categoryattribute` `\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}`

Tests if the category has the given attribute set.

```
3949 \newcommand*\glshascategoryattribute[4]{%
3950   \ifcvoid{@glxtr@categoryattr@@#1@#2}{#4}{#3}%
3951 }
```

`\glssetattribute` `\glssetattribute{<entry label>}{<attribute-label>}{<value>}`

Short cut where the category label is obtained from the entry information.

```
3952 \newcommand*\glssetattribute[3]{%
```

```

3953 \glsssetcategoryattribute{\glscategory{#1}}{#2}{#3}%
3954 }

```

`\glsggetattribute` `\glsggetattribute{<entry label>}{<attribute-label>}`

Short cut where the category label is obtained from the entry information.

```

3955 \newcommand*{\glsggetattribute}[2]{%
3956 \glsssetcategoryattribute{\glscategory{#1}}{#2}%
3957 }

```

`\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.

```

3958 \newcommand*{\glshasattribute}[4]{%
3959 \ifglssentryexists{#1}%
3960 {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
3961 {#4}%
3962 }

```

`categoryattribute` `\glssifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

True if category has the attribute with the given value.

```

3963 \newcommand{\glssifcategoryattribute}[5]{%
3964 \ifcsundef{@glssxtr@categoryattr@#1@#2}%
3965 {#5}%
3966 {\ifcsstring{@glssxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
3967 }

```

`\glssifattribute` `\glssifattribute{<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.

```

3968 \newcommand{\glssifattribute}[5]{%
3969 \ifglssentryexists{#1}%
3970 {\glssifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
3971 {#5}%
3972 }

```

Set attributes for the default general category:

```
3973 \glsssetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
3974 \glsssetcategoryattribute{acronym}{regular}{true}
```

`regularcategory` Convenient shortcut to create add the regular attribute.

```
3975 \newcommand*\glsssetregularcategory}[1]{%
3976   \glsssetcategoryattribute{#1}{regular}{true}%
3977 }
```

`ifregularcategory` `\glssifregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```
3978 \newcommand{\glssifregularcategory}[3]{%
3979   \glssifcategoryattribute{#1}{regular}{true}{#2}{#3}%
3980 }
```

`ifnotregularcategory` `\glssifnotregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```
3981 \newcommand{\glssifnotregularcategory}[3]{%
3982   \glssifcategoryattribute{#1}{regular}{false}{#2}{#3}%
3983 }
```

`\glssifregular` `\glssifregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
3984 \newcommand{\glssifregular}[3]{%
3985   \glssifregularcategory{\glscategory{#1}}{#2}{#3}%
3986 }
```

`\glssifnotregular` `\glssifnotregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
3987 \newcommand{\glssifnotregular}[3]{%
3988   \glssifnotregularcategory{\glscategory{#1}}{#2}{#3}%
3989 }
```

oreachincategory

```
\glsforeachincategory[⟨glossary labels⟩]{⟨category-label⟩}
{⟨glossary-cs⟩}{⟨label-cs⟩}{⟨body⟩}
```

Iterates through all entries in all the glossaries (or just those listed in *⟨glossary labels⟩*) and does *⟨body⟩* if the category matches *⟨category-label⟩*. The control sequences *⟨glossary-cs⟩* and *⟨label-cs⟩* may be used in *⟨body⟩* to access the glossary label and entry label for the current iteration.

```
3990 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
3991   \forallglossaries[#1]{#3}%
3992   {%
3993     \forlgsentries[#3]{#4}%
3994     {%
3995       \glsifcategory{#4}{#2}{#5}{}%
3996     }%
3997   }%
3998 }
```

achwithattribute

```
\glsforeachwithattribute[⟨glossary labels⟩]{⟨attribute-label⟩}
{⟨attribute-value⟩}{⟨glossary-cs⟩}{⟨label-cs⟩}{⟨body⟩}
```

Iterates through all entries in all the glossaries (or just those listed in *⟨glossary labels⟩*) and does *⟨body⟩* if the category attribute *⟨attribute-label⟩* matches *⟨attribute-value⟩*. The control sequences *⟨glossary-cs⟩* and *⟨label-cs⟩* may be used in *⟨body⟩* to access the glossary label and entry label for the current iteration.

```
3999 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
4000   \forallglossaries[#1]{#4}%
4001   {%
4002     \forlgsentries[#4]{#5}%
4003     {%
4004       \glsifattribute{#5}{#2}{#3}{#6}{}%
4005     }%
4006   }%
4007 }
```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to `index` and add `\glstrpostdescription`.

```
4008 \ifdef\newterm
4009 {%
```

`\newterm`

```
4010   \renewcommand*{\newterm}[2][ ]{%
4011     \newglossaryentry{#2}%
4012     {type={index},category=index,name={#2},%
```

```

4013     description={\glxtrpostdescription\nopostdesc},#1}%
4014 }

```

Indexed terms are regular by default.

```

4015 \glissetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

4016 \newcommand*{\glxtrpostdescindex}{}

4017 }
4018 {}

```

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.

```

4019 \ifdef\printsymbols
4020 {%

```

glxtrnewsymbol Unlike \newterm, this has a separate argument for the label (since the symbol will likely contain commands).

```

4021 \newcommand*{\glxtrnewsymbol}[3] [] {%
4022   \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
4023 }

```

Symbols are regular by default.

```

4024 \glissetcategoryattribute{symbol}{regular}{true}

```

rpostdescsymbol

```

4025 \newcommand*{\glxtrpostdescsymbol}{}

4026 }
4027 {}

```

Similar for the numbers option.

```

4028 \ifdef\printnumbers
4029 {%

```

glxtrnewnumber

```

4030 \ifdef\printnumbers
4031 \newcommand*{\glxtrnewnumber}[3] [] {%
4032   \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
4033 }

```

Numbers are regular by default.

```

4034 \glissetcategoryattribute{number}{regular}{true}

```

rpostdescnumber

```

4035 \newcommand*{\glxtrpostdescnumber}{}

```

```
4036 }
4037 {}
```

`\glstrsetcategory` Set the category for all listed labels. The first argument is the list of entry labels and the second argument is the category label.

```
4038 \newcommand*{\glstrsetcategory}[2]{%
4039   \@for\@glstr@label:=#1\do
4040   {%
4041     \glsfieldxdef{\@glstr@label}{category}{#2}%
4042   }%
4043 }
```

`\glstrcategoryforall` 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.

```
4044 \newcommand*{\glstrcategoryforall}[2]{%
4045   \forallglossaries[#1]{\@glstr@type}{%
4046     \forallsentries[\@glstr@type]{\@glstr@label}%
4047     {%
4048       \glsfieldxdef{\@glstr@label}{category}{#2}%
4049     }%
4050   }%
4051 }
```

`\glstrfieldtitlecase` `\glstrfieldtitlecase{\langle label \rangle}{\langle field \rangle}`

Apply title casing to the contents of the given field.

```
4052 \newcommand*{\glstrfieldtitlecase}[2]{%
4053   \expandafter\glstrfieldtitlecasesecs\expandafter
4054   {\csname glo@glsdetoklabel{#1}@#2\endcsname}%
4055 }
```

`\glstrfieldtitlecasesecs` The command used by `\glstrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```
4056 \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.

```
4057 \@ifpackageloaded{glossaries-accsupp}
4058 {
4059   \renewcommand*{\glossentrydesc}[1]{%
4060     \glsdoifexistsorwarn{#1}%
4061     {%
4062       \glssetabbrvfmt{\glscategory{#1}}%
```


As from version 1.04, allow the glossdescfont attribute to determine the font applied.

```

4063 \glshasattribute{#1}{glossdescfont}%
4064 {%
4065 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
4066 \ifcsdef{\@glxtr@attrval}%
4067 {%
4068 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
4069 }%
4070 {%
4071 \GlossariesExtraWarning{Unknown control sequence name
4072 '\@glxtr@attrval' supplied in glossdescfont attribute
4073 for entry '#1'. Ignoring}%
4074 \let\@glxtr@glossdescfont\@firstofone
4075 }%
4076 }%
4077 {\let\@glxtr@glossdescfont\@firstofone}%
4078 \glusifattribute{#1}{glossdesc}{firstuc}%
4079 {%
4080 \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
4081 }%
4082 {%
4083 \glusifattribute{#1}{glossdesc}{title}%
4084 {%
4085 \@glxtr@do@titlecaps@warn
4086 \glsdescriptionaccessdisplay
4087 {%
4088 \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
4089 }%
4090 {#1}%
4091 }%
4092 {%
4093 \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
4094 }%
4095 }%
4096 }%
4097 }
4098 }
4099 {
4100 \renewcommand*\glossentrydesc}[1]{%
4101 \glsdoifexistsorwarn{#1}%
4102 {%
4103 \glissetabbrvfmt{\glscategory{#1}}%
4104 \glshasattribute{#1}{glossdescfont}%
4105 {%
4106 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
4107 \ifcsdef{\@glxtr@attrval}%
4108 {%
4109 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
4110 }%

```

```

4111      {%
4112      \GlossariesExtraWarning{Unknown control sequence name
4113      '\@glsxtr@attrval' supplied in glossdescfont attribute
4114      for entry '#1'. Ignoring}%
4115      \let\@glsxtr@glossdescfont\@firstofone
4116      }%
4117  }%
4118  {\let\@glsxtr@glossdescfont\@firstofone}%
4119  \glsifattribute{#1}{glossdesc}{firstuc}%
4120  {%
4121      \@glsxtr@glossdescfont{\Glsentrydesc{#1}}%
4122  }%
4123  {%
4124      \glsifattribute{#1}{glossdesc}{title}%
4125      {%
4126          \@glsxtr@do@titlecaps@warn
4127          \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
4128      }%
4129      {%
4130          \@glsxtr@glossdescfont{\Glsentrydesc{#1}}%
4131      }%
4132  }%
4133  }%
4134  }
4135  }

```

`\glossentryname` If the glossname attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

4136 \ifpackageloaded{glossaries-accsupp}
4137 {
4138   \renewcommand*{\glossentryname}[1]{%
4139     \@glsdoifexistsorwarn{#1}%
4140     {%
4141       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

4142   \glshasattribute{#1}{glossnamefont}%
4143   {%
4144     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
4145     \ifcsdef{\@glsxtr@attrval}%
4146     {%
4147       \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
4148     }%
4149     {%
4150       \GlossariesExtraWarning{Unknown control sequence name
4151       '\@glsxtr@attrval' supplied in glossnamefont attribute
4152       for entry '#1'. Reverting to default \string\glsnamefont}%
4153       \let\@glsxtr@glossnamefont\glsnamefont
4154     }%
4155   }%

```

```

4156     {\let\@glsxtr@glossnamefont\glsnamefont}%
4157     \glsifattribute{#1}{glossname}{firstuc}%
4158     {%
4159         \glsnameaccessdisplay
4160         {%
4161             \@glsxtr@glossnamefont{\Glsentryname{#1}}%
4162             }%
4163             {#1}%
4164         }%
4165         {%
4166             \glsifattribute{#1}{glossname}{title}%
4167             {%
4168                 \@glsxtr@do@titlecaps@warn
4169                 \glsnameaccessdisplay
4170                 {%
4171                     \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
4172                     }%
4173                     {#1}%
4174                 }%
4175                 {%
4176                     \glsifattribute{#1}{glossname}{uc}%
4177                     {%
4178                         \glsnameaccessdisplay
4179                         {%

```

Hide the label from the upper-casing command.

```

4180             \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
4181             \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
4182             }%
4183             {#1}%
4184         }%
4185         {%
4186             \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
4187             \glsnameaccessdisplay
4188             {%
4189                 \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
4190                 }%
4191                 {#1}%
4192             }%
4193         }%
4194     }%

```

Do post-name hook:

```

4195     \glsxtrpostnamehook{#1}%
4196     }%
4197 }
4198 }
4199 {
4200     \renewcommand*{\glossentryname}[1]{%
4201         \@glsdoifexistsorwarn{#1}%

```

```

4202 {%
4203     \glsetabbrvfmt{\glscategory{#1}}%
4204     \glshasattribute{#1}{glossnamefont}%
4205     {%
4206         \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
4207         \ifcsdef{\@glxtr@attrval}%
4208         {%
4209             \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
4210         }%
4211         {%
4212             \GlossariesExtraWarning{Unknown control sequence name
4213             '\@glxtr@attrval' supplied in glossnamefont attribute
4214             for entry '#1'. Reverting to default \string\glsgnamefont}%
4215             \let\@glxtr@glossnamefont\glsgnamefont
4216         }%
4217     }%
4218     {\let\@glxtr@glossnamefont\glsgnamefont}%
4219     \glsgifattribute{#1}{glossname}{firstuc}%
4220     {%
4221         \@glxtr@glossnamefont{\Glsentryname{#1}}%
4222     }%
4223     {%
4224         \glsgifattribute{#1}{glossname}{title}%
4225         {%
4226             \@glxtr@do@titlecaps@warn
4227             \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
4228         }%
4229         {%
4230             \glsgifattribute{#1}{glossname}{uc}%
4231         }%

```

Hide the label from the upper-casing command.

```

4232         \letcs{\glo@name}{\glo@glsgdetoklabel{#1}@name}%
4233         \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
4234     }%
4235     {%

```

This little trick is used by glossaries to allow the user to redefine \glsgnamefont to use \makefirstuc. Support it even though they can now use the firstuc attribute.

```

4236         \letcs{\glo@name}{\glo@glsgdetoklabel{#1}@name}%
4237         \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
4238     }%
4239 }%
4240 }%

```

Do post-name hook.

```

4241     \glxtrpostnamehook{#1}%
4242 }%
4243 }
4244 }

```

`\Glossentryname` Redefine to set the abbreviation format and accessibility support.

```
4245 \ifpackageloaded{glossaries-accsupp}
4246 {
4247   \renewcommand*{\Glossentryname}[1]{%
4248     \@glsdoifexistsorwarn{#1}%
4249     {%
4250       \glsetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```
4251     \glshasattribute{#1}{glossnamefont}%
4252     {%
4253       \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
4254       \ifcsdef{\@glstr@attrval}%
4255       {%
4256         \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
4257       }%
4258       {%
4259         \GlossariesExtraWarning{Unknown control sequence name
4260           '\@glstr@attrval' supplied in glossnamefont attribute
4261           for entry '#1'. Reverting to default \string\glnamefont}%
4262         \let\@glstr@glossnamefont\glnamefont
4263       }%
4264     }%
4265     {\let\@glstr@glossnamefont\glnamefont}%
4266     \glnameaccessdisplay
4267     {%
4268       \@glstr@glossnamefont{\Glsentryname{#1}}%
4269     }%
4270     {#1}%
```

Do post-name hook:

```
4271     \glstrpostnamehook{#1}%
4272   }%
4273 }
4274 }
4275 {
4276   \renewcommand*{\Glossentryname}[1]{%
4277     \@glsdoifexistsorwarn{#1}%
4278     {%
4279       \glsetabbrvfmt{\glscategory{#1}}%
4280       \glshasattribute{#1}{glossnamefont}%
4281       {%
4282         \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
4283         \ifcsdef{\@glstr@attrval}%
4284         {%
4285           \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
4286         }%
4287         {%
4288           \GlossariesExtraWarning{Unknown control sequence name
4289             '\@glstr@attrval' supplied in glossnamefont attribute
```

```

4290         for entry ‘#1’. Reverting to default \string\glsnamefont}%
4291         \let\@glsxtr@glossnamefont\glsnamefont
4292     }%
4293 }%
4294 {\let\@glsxtr@glossnamefont\glsnamefont}%
4295 \@glsxtr@glossnamefont{\Glsentryname{#1}}}%

```

Do post-name hook:

```

4296     \glsxtrpostnamehook{#1}%
4297 }%
4298 }
4299 }

```

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.

`xtrpostnamehook` Hook to append stuff after the name is displayed in the glossary. The argument is the entry's label.

```

4300 \newcommand*{\glsxtrpostnamehook}[1]{%
4301   \def\@glsnumberformat{glsnumberformat}%
4302   \glsxtrdoautoindexname{#1}{indexname}%

```

Allow categories to hook in here.

```

4303   \csuse{glsxtrpostname\glscategory{\glscurrententrylabel}}%
4304 }

```

`format@override` Determines if the format key should override the indexing attribute value.

```

4305 \newif\ifglsxtr@format@override
4306 \@glsxtr@format@overridefalse

```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

4307 \@ifpackageloaded{hyperref}
4308 {

```

If `hyperref`'s `hyperindex` option is on, then `hyperref` will automatically add `\hyperpage`, so don't add it.

```

4309   \ifHy@hyperindex
4310     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
4311       \@glsxtr@format@override true
4312       \appto\theindex{\let\glshypernumber\@firstofone}%
4313     }
4314   \else
4315     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
4316       \@glsxtr@format@override true
4317       \appto\theindex{\let\glshypernumber\hyperpage}%
4318     }
4319   \fi

```

```

4320 }
4321 {
4322   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
4323     \@glstr@format@overridetrue
4324   }
4325 }
4326 \onlypreamble\GlsXtrEnableIndexFormatOverride

```

doautoindexname

```

4327 \newcommand*{\glstrdoautoindexname}[2]{%
4328   \glshasattribute{#1}{#2}%
4329   {%

```

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.

```

4330   \@glstr@autoindex@setname{#1}%

```

If the attribute value is simply “true” don't add an encap, otherwise use the value as the encap.

```

4331   \protected@edef\@glstr@attrval{\glsggetattribute{#1}{#2}}%
4332   \if@glstr@format@override
4333     \ifdefstring{\@glstrnumberformat}{\glstrnumberformat}{}%
4334     {\let\@glstr@attrval\@glstrnumberformat}%
4335   \fi
4336   \ifdefstring{\@glstr@attrval}{true}%
4337   {%
4338     {\eappto\@glo@name{\@glstr@autoindex@encap\@glstr@attrval}}%
4339     \expandafter\index\expandafter{\@glo@name}%
4340   }%
4341   {}%
4342 }

```

toindex@setname Assign \@glo@name for use with indexname attribute.

```

4343 \newcommand*{\@glstr@autoindex@setname}[1]{%
4344   \def\@glo@name{\string\glstryname{#1}}%
4345   \glstryentryfield{\@glo@sort}{#1}{sort}%
4346   \@glstr@checkmkidxchars\@glo@sort
4347   \@glstr@autoindex@doextra@esc\@glo@sort
4348   \epreto\@glo@name{\@glo@sort\@glstr@autoindex@at}%
4349 }

```

dex@doextra@esc

```

4350 \newcommand*{\@glstr@autoindex@doextra@esc}[1]{%

```

Escape the escape character unless it has already been escaped.

```

4351   \ifx\@glstr@autoindex@esc\@glstr@quotechar
4352   \else
4353     \def\@glstr@checkedmkidx{%
4354       \edef\@glstr@checkspch{%
4355         \noexpand\@glstr@autoindex@escquote\expandonce{#1}%
4356         \noexpand\@empty\@glstr@autoindex@esc\noexpand\@nnil

```

```

4357      \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4358      \@glsxtr@checkspch
4359      \let#1\@gls@checkedmkidx\relax
4360 \fi

```

Escape actual character unless it has already been escaped.

```

4361 \ifx\@glsxtr@autoindex@at\@gls@actualchar
4362 \else
4363   \def\@gls@checkedmkidx{%
4364     \edef\@glsxtr@checkspch{%
4365       \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
4366       \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
4367       \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4368     \@glsxtr@checkspch
4369     \let#1\@gls@checkedmkidx\relax
4370 \fi

```

Escape level character unless it has already been escaped.

```

4371 \ifx\@glsxtr@autoindex@level\@gls@levelchar
4372 \else
4373   \def\@gls@checkedmkidx{%
4374     \edef\@glsxtr@checkspch{%
4375       \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
4376       \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
4377       \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4378     \@glsxtr@checkspch
4379     \let#1\@gls@checkedmkidx\relax
4380 \fi

```

Escape encap character unless it has already been escaped.

```

4381 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
4382 \else
4383   \def\@gls@checkedmkidx{%
4384     \edef\@glsxtr@checkspch{%
4385       \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
4386       \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
4387       \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4388     \@glsxtr@checkspch
4389     \let#1\@gls@checkedmkidx\relax
4390 \fi
4391 }

```

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`.

```

4392 \newcommand*{\@glsxtr@autoindex@at}{}

```

`\trSetActualChar` Set the actual character.

```

4393 \newcommand*{\GlsXtrSetActualChar}[1]{%

```



```

4394 \gdef\@glsxtr@autoindex@at{#1}%
4395 \def\@glsxtr@autoindex@escat##1#1##2#1##3\@glsxtr@endescspch{%
4396 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escat}{##1}{##2}{##3}%
4397 }%
4398 }
4399 \@onlypreamble\GlsXtrSetActualChar
4400 \makeatother
4401 \GlsXtrSetActualChar{@}
4402 \makeatletter

```

autoindex@encap Encap character for use with `\index`.

```
4403 \newcommand*{\@glsxtr@autoindex@encap}{}
```

XtrSetEncapChar Set the encap character.

```

4404 \newcommand*{\GlsXtrSetEncapChar}[1]{%
4405 \gdef\@glsxtr@autoindex@encap{#1}%
4406 \def\@glsxtr@autoindex@escencap##1#1##2#1##3\@glsxtr@endescspch{%
4407 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escencap}{##1}{##2}{##3}%
4408 }%
4409 }
4410 \GlsXtrSetEncapChar{||}
4411 \@onlypreamble\GlsXtrSetEncapChar

```

autoindex@level Level character for use with `\index`.

```
4412 \newcommand*{\@glsxtr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```

4413 \newcommand*{\GlsXtrSetLevelChar}[1]{%
4414 \gdef\@glsxtr@autoindex@level{#1}%
4415 \def\@glsxtr@autoindex@esclevel##1#1##2#1##3\@glsxtr@endescspch{%
4416 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@esclevel}{##1}{##2}{##3}%
4417 }%
4418 }
4419 \GlsXtrSetLevelChar{!}
4420 \@onlypreamble\GlsXtrSetLevelChar

```

r@autoindex@esc Escape character for use with `\index`.

```
4421 \newcommand*{\@glsxtr@autoindex@esc}{""}
```

lsXtrSetEscChar Set the escape character.

```

4422 \newcommand*{\GlsXtrSetEscChar}[1]{%
4423 \gdef\@glsxtr@autoindex@esc{#1}%
4424 \def\@glsxtr@autoindex@escquote##1#1##2#1##3\@glsxtr@endescspch{%
4425 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escquote}{##1}{##2}{##3}%
4426 }%
4427 }
4428 \GlsXtrSetEscChar{"}
4429 \@onlypreamble\GlsXtrSetEscChar

```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```
4430 \ifdef\actualchar
4431 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
4432 {}}
```

Quote character \quotechar:

```
4433 \ifdef\quotechar
4434 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
4435 {}}
```

Level character \levelchar:

```
4436 \ifdef\levelchar
4437 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
4438 {}}
```

Encap character \encapchar:

```
4439 \ifdef\encapchar
4440 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
4441 {}}
```

leto@endescspch

```
4442 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}}
```

toindex@esc@spch

```
\@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}}
```

```
4443 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
4444   \@glstmpb=\expandafter{\@glsc@checkedmkidx}%
4445   \toks@={#3}%
4446   \ifx\@nnil#3\relax
4447     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
4448   \else
4449     \ifx\@nnil#4\relax
4450       \edef\@glsc@checkedmkidx{\the\@glstmpb\the\toks@}%
4451       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
4452         #4#5\@glxtr@endescspch}%
4453     \else
4454       \edef\@glsc@checkedmkidx{\the\@glstmpb\the\toks@
4455         \@glxtr@autoindex@esc#1}%
4456       \def\@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
4457     \fi
4458   \fi
4459   \@glxtr@checkspch
4460 }
```

\Glossentrydesc Redefine to set the abbreviation format and accessibility support.

```
4461 \renewcommand*{\Glossentrydesc}[1]{%
4462   \glsoifexistsorwarn{#1}%
```

```

4463 {%
4464   \glssetabbrvfmt{\glscategory{#1}}%
4465   \Glsaccessdesc{#1}%
4466 }%
4467 }

```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```

4468 \renewcommand*{\glossentrysymbol}[1]{%
4469   \glsdoifexistsorwarn{#1}%
4470   {%
4471     \glssetabbrvfmt{\glscategory{#1}}%
4472     \glsaccesssymbol{#1}%
4473   }%
4474 }

```

`\Glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```

4475 \renewcommand*{\Glossentrysymbol}[1]{%
4476   \glsdoifexistsorwarn{#1}%
4477   {%
4478     \glssetabbrvfmt{\glscategory{#1}}%
4479     \Glsaccesssymbol{#1}%
4480   }%
4481 }

```

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.

```

4482 \newcommand*{\GlsXtrEnableInitialTagging}{%
4483   \@ifstar\s@glsextr@enabletagging\@glsextr@enabletagging
4484 }
4485 \@onlypreamble\GlsXtrEnableInitialTagging

```

`\r@enabletagging` Starred version undefines command.

```

4486 \newcommand*{\s@glsextr@enabletagging}[2]{%
4487   \undef#2%
4488   \@glsextr@enabletagging{#1}{#2}%
4489 }

```

`\r@enabletagging` Internal command.

```

4490 \newcommand*{\@glsextr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
4491   \@for\@glsextr@cat:=#1\do
4492   {%
4493     \ifdefempty\@glsextr@cat
4494     {}%
4495     {\glssetcategoryattribute{\@glsextr@cat}{tagging}{true}}%

```

```

4496 }%
4497 \newrobustcmd*#2[1]{##1}%
4498 \def\@glxstr@taggingcs{#2}%
4499 \renewcommand*\@glxstr@activate@initialtagging{%
4500   \let#2\@glxstr@tag
4501 }%
4502 \ifundef\@gls@preglossaryhook
4503 {\GlossariesExtraWarning{Initial tagging requires at least
4504   glossaries.sty v4.19 to work correctly}}%
4505 {}%
4506 }

```

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

```

4507 \ifundef\mfu@checkword@do
4508 {
4509   \newcommand*\mfu@checkword@do[1]{%
4510     \ifdefstring{\mfu@checkword@arg}{#1}%
4511     {%
4512       \let\@mfu@domakefirstuc\@firstofone
4513       \listbreak
4514     }%
4515   }%
4516 }

```

`\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.

```

4517 \ifundef\mfu@checkword
4518 {
4519   \newcommand{\@glxstr@do@titlecaps@warn}{%
4520     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
4521       support not available}%

```

One warning should suffice.

```

4522     \let\@glxstr@do@titlecaps@warn\relax
4523   }
4524 }
4525 {
4526   \renewcommand*\mfu@checkword[1]{%
4527     \def\mfu@checkword@arg{#1}%
4528     \let\@mfu@domakefirstuc\makefirstuc
4529     \forlistloop\mfu@checkword@do\@mfu@nocaplist
4530   }
4531 }
4532 }
4533 {}% no patch required

```

`@titlecaps@warn` Do warning if title case not supported.

4534 \newcommand*{\@glxtr@do@titlecaps@warn}{}

@initialtagging Used in \printglossary but at least v4.19 of glossaries required.

4535 \newcommand*\@glxtr@activate@initialtagging{}

\@glxtr@tag Definition of tagging command when used in glossary.

4536 \newrobustcmd*{\@glxtr@tag}[1]{%

4537 \gl@ifattribute{\glscurrententrylabel}{tagging}{true}%

4538 {\glxtrtagfont{#1}}{#1}%

4539 }

\glxtrtagfont Used in the glossary.

4540 \newcommand*\glxtrtagfont[1]{\underline{#1}}

preglossaryhook This macro was introduced in glossaries version 4.19, so it may not be defined. If it hasn't been defined this feature is unavailable. A check is added for the entry's existence to prevent errors from occurring if the user removes an entry or changes the label, which can interrupt the build process.

4541 \ifdef\@gls@preglossaryhook

4542 {

4543 \renewcommand*\@gls@preglossaryhook{%

4544 \@glxtr@activate@initialtagging

Since the glossaries are automatically scoped, \@glxtr@org@postdescription shouldn't already be defined, but check anyway just as a precautionary measure.

4545 \ifundef\@glxtr@org@postdescription

4546 {%

4547 \let\@glxtr@org@postdescription\glspostdescription

4548 \renewcommand*\glspostdescription{%

4549 \ifglentryexists{\glscurrententrylabel}%

4550 {%

4551 \glxtrpostdescription

4552 \@glxtr@org@postdescription

4553 }%

4554 {}}%

4555 }%

4556 }%

4557 {}%

Enable the options used by \@@glxtrp:

4558 \glossxtrsetpopts

4559 }%

4560 }

4561 {}

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.

```

4562 \newcommand*{\glxtrpostdescription}{%
4563   \csuse{glxtrpostdesc\glscategory{\glscurrententrylabel}}%
4564 }

postdescgeneral
4565 \newcommand*{\glxtrpostdescgeneral}{}

xtrpostdescterm
4566 \newcommand*{\glxtrpostdescterm}{}

postdescacronym
4567 \newcommand*{\glxtrpostdescacronym}{}

escabbreviation
4568 \newcommand*{\glxtrpostdescabbreviation}{}

glspostlinkhook  Redefine the post link hook used by commands like \gls to make it easier for categories
                  or attributes to modify this action. Since this hook occurs outside the existence check of
                  commands like \gls, this needs to be checked again here. Do nothing if the entry hasn't
                  been defined.
4569 \renewcommand*{\glspostlinkhook}{%
4570   \ifglseentryexists{\glslabel}{\glxtrpostlinkhook}{}%
4571 }

xtrpostlinkhook  The entry label should already be stored in \glslabel by \@gls@link.
4572 \newcommand*{\glxtrpostlinkhook}{%
4573   \glxtrdiscardperiod{\glslabel}%
4574   {\glxtrpostlinkendsentence}%
4575   {\glxtrpostlink}%
4576 }

\glxtrpostlink
4577 \newcommand*{\glxtrpostlink}{%
4578   \csuse{glxtrpostlink\glscategory{\glslabel}}%
4579 }

linkendsentence  Done by \glxtrpostlinkhook if a full stop is discarded.
4580 \newcommand*{\glxtrpostlinkendsentence}{%
4581   \ifcsdef{glxtrpostlink\glscategory{\glslabel}}
4582   {%
4583     \csuse{glxtrpostlink\glscategory{\glslabel}}%
    Put the full stop back.
4584     .\spacefactor\sffcode'\. \relax
4585   }%
4586   {%

```

Assume the full stop was discarded because the entry ends with a period, so adjust the space-factor.

```
4587 \spacefactor\sfcode'\. \relax
4588 }%
4589 }
```

addDescOnFirstUse 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.

```
4590 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
4591 \glxtrifwasfirstuse{\space(\glssaccessdesc{\glslabel})}{}%
4592 }
```

addSymbolOnFirstUse 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.

```
4593 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
4594 \glxtrifwasfirstuse
4595 {%
4596 \ifglshassymbol{\glslabel}{\space(\glssaccesssymbol{\glslabel})}{}%
4597 }%
4598 {}%
4599 }
```

discardperiod 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).

```
4600 \newcommand*{\glxtrdiscardperiod}[3]{%
4601 \glxtrifwasfirstuse
4602 {%
4603 \glssifattribute{#1}{retainfirstuseperiod}{true}%
4604 {#3}%
4605 {%
4606 \glssifattribute{#1}{discardperiod}{true}%
4607 {%
4608 \glssifplural
4609 {%
4610 \glssifattribute{#1}{pluraldiscardperiod}{true}%
4611 {\glxtrifperiod{#2}{#3}}%
4612 {#3}%
4613 }%
4614 }%
4615 \glxtrifperiod{#2}{#3}%
4616 }%
4617 }%
4618 {#3}%
4619 }%
4620 }%
4621 {%
```

```

4622 \glsifattribute{#1}{discardperiod}{true}%
4623 {%
4624   \glsifplural
4625   {%
4626     \glsifattribute{#1}{pluraldiscardperiod}{true}%
4627     {\glsxtrifperiod{#2}{#3}}%
4628     {#3}%
4629   }%
4630   {%
4631     \glsxtrifperiod{#2}{#3}%
4632   }%
4633   }%
4634   {#3}%
4635 }%
4636 }

```

`\glsxtrifperiod` Make a convenient user command to check if the next character is a full stop (period). Works like `\@ifstar` but uses `\new@ifnextchar` rather than `\@ifnextchar`

```
4637 \newcommand*{\glsxtrifperiod}[1]{\new@ifnextchar.\@firstoftwo{#1}}
```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`\glsxtr@punclist` List of characters identified as punctuation marks. (Be careful of babel shorthands!) This doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```
4638 \newcommand*{\glsxtr@punclist}{.,,:;?!}
```

`\punctuationmark` Add character to punctuation list.

```
4639 \newcommand*{\glsxtraddpunctuationmark}[1]{\appto\glsxtr@punclist{#1}}
```

`\punctuationmarks` Reset the punctuation list.

```
4640 \newcommand*{\glsxtrsetpunctuationmarks}[1]{\def\glsxtr@punclist{#1}}
```

`\glsxtrifpunc` `\glsxtrifnextpunc{<true part>}{<>false part>}`

Test if this is followed by a punctuation mark. (Adapted from `\new@ifnextchar`.)

```

4641 \newcommand*{\glsxtrifnextpunc}[2]{%
4642   \def\reserved@a{#1}%
4643   \def\reserved@b{#2}%
4644   \futurelet\@glsxpunc@token\glsxtr@ifnextpunc
4645 }

```

`\glsxtr@ifnextpunc`

```

4646 \newcommand*{\glsxtr@ifnextpunc}{%
4647   \glsxtr@ifpunctoken{\@glsxpunc@token}{\let\reserved@b\reserved@a}{}%
4648   \reserved@b
4649 }

```


xtr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
4650 \newcommand*{\glxtr@ifpunctoken}[1]{%
4651   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
4652 }
```

xtr@ifpunctoken

```
4653 \def\@glxtr@ifpunctoken#1#2{%
4654   \let\reserved@d=#2%
4655   \ifx\reserved@d\@nnil
4656     \let\glxtr@next\@glxtr@notfoundinlist
4657   \else
4658     \ifx#1\reserved@d
4659       \let\glxtr@next\@glxtr@foundinlist
4660     \else
4661       \let\glxtr@next\@glxtr@ifpunctoken
4662     \fi
4663   \fi
4664   \glxtr@next#1%
4665 }
```

xtr@foundinlist

```
4666 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
4667 \def\@glxtr@notfoundinlist#1{\@secondoftwo}
```

glxtrdopostpunc

`\glxtrdopostpunc{<code>}`

If this is followed by a punctuation character, do *<code>* after the character otherwise do *<code>* before whatever comes next.

```
4668 \newcommand{\glxtrdopostpunc}[1]{%
4669   \glxtr@ifnextpunc{\@glxtr@swaptwo{#1}}{#1}%
4670 }
```

@glxtr@swaptwo

```
4671 \newcommand{\@glxtr@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.

```

4672 \define@key{glsxtrabbrv}{category}{%
4673   \edef\glscategorylabel{#1}%
4674   \ifcsdef{@glsabbrv@current@#1}%
4675   {%

```

Warning should already have been issued.

```

4676   \let\@glsxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
4677   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
4678   \glsxtr@applyabbrvstyle{\csname @glsabbrv@current@#1\endcsname}%
4679   \let\GlsXtrWarnDeprecatedAbbrStyle\@glsxtr@orgwarndep
4680 }%
4681 {}%
4682 }

```

Save the short plural form. This may be needed before the entry is defined.

```

4683 \define@key{glsxtrabbrv}{shortplural}{%
4684   \def\@gls@shortpl{#1}%
4685 }

```

Similarly for the long plural form.

```

4686 \define@key{glsxtrabbrv}{longplural}{%
4687   \def\@gls@longpl{#1}%
4688 }

```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```

4689 \newtoks\glsshortpltok

```

\glslongpltok

```

4690 \newtoks\glslongpltok

```

glsxtr@insertdots Provided in case user wants to automatically insert dots between each letter of the abbreviation. This should be applied before defining the abbreviation to optimise the document build. (Otherwise, it would have to be done each time the short form is required, which is an unnecessary waste of time.) For this to work the short form must be expanded when passed to \newabbreviation. Note that explicitly using the short or shortplural keys will override this.

```

4691 \newcommand*{\@glsxtr@insertdots}[2]{%
4692   \def#1{%
4693     \@glsxtr@insert@dots#1#2\@nnil
4694   }

```

glsxtr@insert@dots

```

4695 \newcommand*{\@glsxtr@insert@dots}[2]{%
4696   \ifx\@nnil#2\relax
4697   \let\@glsxtr@insert@dots@next\@gobble
4698   \else
4699   \ifx\relax#2\relax

```

```

4700 \else
4701 \appto#1{#2.}%
4702 \fi
4703 \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
4704 \fi
4705 \@glsxtr@insert@dots@next#1%
4706 }

```

newabbreviation Define a new generic abbreviation.

```

4707 \newcommand*{\newabbreviation}[4][]{%
4708 \glsxtr@newabbreviation{#1}{#2}{#3}{#4}%
4709 }

```

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.)

```

4710 \newcommand*{\glsxtr@newabbreviation}[4]{%
4711 \glskeylisttok{#1}%
4712 \glslabeltok{#2}%
4713 \glsshorttok{#3}%
4714 \glslongtok{#4}%

```

Get the category.

```

4715 \def\glscategorylabel{abbreviation}%
4716 \glsxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
4717 \setkeys*{glsxtrabbrv}{shortplural,longplural}{#1}%

```

Set the default long plural

```

4718 \def\@gls@longpl{#4\glspluralsuffix}%

```

Has the insertdots attribute been set?

```

4719 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
4720 {%
4721 \@glsxtr@insertdots\@gls@short{#3}%
4722 \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
4723 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
4724 {%
4725 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
4726 '\abbrvpluralsuffix}%
4727 }%
4728 {%
4729 \glsifcategoryattribute{\glscategorylabel}{nosshortplural}{true}%
4730 {%
4731 \let\@gls@shortpl\@gls@short
4732 }%
4733 {%
4734 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
4735 \abbrvpluralsuffix}%
4736 }%
4737 }%
4738 }%
4739 {%

```

insertdots not true.

```
4740 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
4741 {%
4742   \def\@gls@shortpl{#3'\abbrvpluralsuffix}%
4743 }%
4744 {%
4745   \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
4746   {%
4747     \def\@gls@shortpl{#3}%
4748   }%
4749   {%
4750     \def\@gls@shortpl{#3\abbrvpluralsuffix}%
4751   }%
4752 }%
4753 }%
```

Hook for further customisation if required:

```
4754 \glsxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%
```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary.

```
4755 \setkeys*{glsxtrabbrv}[category]{#1}%
```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```
4756 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
4757 \expandafter\glslongpltok\expandafter{\@gls@longpl}%
```

Do any extra setup provided by hook:

```
4758 \newabbreviationhook
```

Define this entry:

```
4759 \protected@edef\@do@newglossaryentry{%
4760   \noexpand\newglossaryentry{\the\glslabeltok}%
4761   {%
4762     type=\glsxtrabbrvtype,%
4763     category=abbreviation,%
4764     short={\the\glsshorttok},%
4765     shortplural={\the\glsshortpltok},%
4766     long={\the\glslongtok},%
4767     longplural={\the\glslongpltok},%
4768     name={\the\glsshorttok},%
4769     \CustomAbbreviationFields,%
4770     \the\glskeylisttok
4771   }%
4772 }%
4773 \@do@newglossaryentry
4774 \GlsXtrPostNewAbbreviation
4775 }
```

evpresetkeyhook Hook for extra stuff in \newabbreviation

```
4776 \newcommand*{\glsxtrnewabbrevpresetkeyhook}[3]{}
```

NewAbbreviation Hook used by abbreviation styles.

```
4777 \newcommand*{\GlsXtrPostNewAbbreviation}{}
```

bbreviationhook Hook for use with \newabbreviation.

```
4778 \newcommand*{\newabbreviationhook}{}%
```

reviationFields

```
4779 \newcommand*{\CustomAbbreviationFields}{}%
```

lsxtrfullformat Full format without case change.

```
4780 \newcommand*{\glsxtrfullformat}[2]{%
4781   \glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
4782   (\protect\glsfirstabbrvfont{\glsaccessshort{#1}})%
4783 }
```

lsxtrfullformat Full format with case change.

```
4784 \newcommand*{\Glsxtrfullformat}[2]{%
4785   \glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
4786   (\protect\glsfirstabbrvfont{\Glsaccessshort{#1}})%
4787 }
```

xtrfullplformat Plural full format without case change.

```
4788 \newcommand*{\glsxtrfullplformat}[2]{%
4789   \glsfirstlongfont{\glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
4790   (\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}})%
4791 }
```

xtrfullplformat Plural full format with case change.

```
4792 \newcommand*{\Glsxtrfullplformat}[2]{%
4793   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
4794   (\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}})%
4795 }
```

\glsxtrfullsep Separator used by full format is a space by default. The argument is the entry's label.

```
4796 \newcommand*{\glsxtrfullsep}[1]{\space}
```

In-line formats in case first use isn't compatible with \glsentryfull (for example, first use suppresses the long form or uses a footnote).

nlinefullformat Full format without case change.

```
4797 \newcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}
```

nlinefullformat Full format with case change.

```
4798 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}
```

xtrfullplformat Plural full format without case change.

```
4799 \newcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}
```

inlinefullplformat Plural full format with case change.
4800 \newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}

Redefine \glsentryfull etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the \glsxtrfull set of commands instead.

\glsentryfull
4801 \renewcommand*{\glsentryfull}[1]{\glsxtrinlinefullformat{#1}{}}

\Glsentryfull
4802 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}

\glsentryfullpl
4803 \renewcommand*{\glsentryfullpl}[1]{\glsxtrinlinefullplformat{#1}{}}

\Glsentryfullpl
4804 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}

gsfirstabbrvfont Font changing command used for the abbreviation on first use or in the full format.
4805 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}

gsabbrvdefaultfont Font changing command used for the abbreviation on first use or in the full format.
4806 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}

\glsabbrvfont Font changing command used for the abbreviation on subsequent use.
4807 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}

gsabbrvdefaultfont
4808 \newcommand*{\glsabbrvdefaultfont}[1]{#1}

\glslongfont Font changing command used for the long form in commands like \glsxtrlong.
4809 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}

gslongdefaultfont Default font changing command used for the long form in commands like \glsxtrlong.
4810 \newcommand*{\glslongdefaultfont}[1]{#1}

glsfirstlongfont Font changing command used for the long form on first use or in the full format.
4811 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}

gslongdefaultfont
4812 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}

gsbrvpluralsuffix Default plural suffix. Allow an alternative default suffix for abbreviations.
4813 \newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}

brvpluralsuffix Default plural suffix.

```
4814 \newcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}
```

\glxtrfull Full form (no case-change).

```
4815 \newrobustcmd*{\glxtrfull}{\@gls@hyp@opt\ns@glxtrfull}
4816 \newcommand*\ns@glxtrfull[2][\%
4817   \new@ifnextchar[\@glxtr@full{#1}{#2}]{%
4818     {\@glxtr@full{#1}{#2}[]}%
4819 }
```

\@glxtr@full Low-level macro:

```
4820 \def\@glxtr@full#1#2[#3]{%
4821   \glsdoifexists{#2}%
4822   {%
4823     \glssetabbrvfmt{\glscategory{#2}}%
4824     \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
4825     \let\glsifplural\@secondoftwo
4826     \let\glscapscase\@firstofthree
4827     \let\glsinsert\@empty
4828     \def\glscustomtext{\glxtrinlinefullformat{#2}{#3}}%

```

What should \glxtrifwasfirstuse be set to here? Where the inline and display full forms are the same, this is essentially emulating first use, to it make sense for the postlink hook to pretend it was a first use instance. It makes less sense if the inline and display forms are different. Provide a hook to make it easier to reconfigure.

```
4829   \glxtrsetupfulldefs
4830   \@gl@s@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4831   }%
4832   \glspostlinkhook
4833 }
```

trsetupfulldefs

```
4834 \newcommand*{\glxtrsetupfulldefs}{%
4835   \let\glxtrifwasfirstuse\@firstoftwo
4836 }
```

\Glsxtrfull Full form (first letter uppercase).

```
4837 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\ns@Glsxtrfull}
4838 \newcommand*\ns@Glsxtrfull[2][\%
4839   \new@ifnextchar[\@Glsxtr@full{#1}{#2}]{%
4840     {\@Glsxtr@full{#1}{#2}[]}%
4841 }
```

\@Glsxtr@full Low-level macro:

```
4842 \def\@Glsxtr@full#1#2[#3]{%
4843   \glsdoifexists{#2}%
4844   {%
4845     \glssetabbrvfmt{\glscategory{#2}}%

```

```

4846 \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
4847 \let\gl@sifplural\@secondoftwo
4848 \let\gl@scapscase\@secondofthree
4849 \let\gl@sininsert\@empty
4850 \def\gl@scustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
4851 \glxtrsetupfulldefs
4852 \@gl@s@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4853 }%
4854 \glspostlinkhook
4855 }

```

\Glsxtrfull Full form (all uppercase).

```

4856 \newrobustcmd*{\Glsxtrfull}{\@gl@s@hyp@opt\@ns@Glsxtrfull}
4857 \newcommand*\ns@Glsxtrfull[2][]{%
4858 \new@ifnextchar[\@Glsxtr@full{#1}{#2}}%
4859 {\@Glsxtr@full{#1}{#2}[]}%
4860 }

```

\@Glsxtr@full Low-level macro:

```

4861 \def\@Glsxtr@full#1#2[#3]{%
4862 \gl@doifexists{#2}%
4863 {%
4864 \gl@setabbrvfmt{\gl@category{#2}}%
4865 \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
4866 \let\gl@sifplural\@secondoftwo
4867 \let\gl@scapscase\@thirdofthree
4868 \let\gl@sininsert\@empty
4869 \def\gl@scustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
4870 \glxtrsetupfulldefs
4871 \@gl@s@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4872 }%
4873 \glspostlinkhook
4874 }

```

\glxtrfullpl Plural full form (no case-change).

```

4875 \newrobustcmd*{\glxtrfullpl}{\@gl@s@hyp@opt\@ns@glxtrfullpl}
4876 \newcommand*\ns@glxtrfullpl[2][]{%
4877 \new@ifnextchar[\@glxtr@fullpl{#1}{#2}}%
4878 {\@glxtr@fullpl{#1}{#2}[]}%
4879 }

```

\@glxtr@fullpl Low-level macro:

```

4880 \def\@glxtr@fullpl#1#2[#3]{%
4881 \gl@doifexists{#2}%
4882 {%
4883 \gl@setabbrvfmt{\gl@category{#2}}%
4884 \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
4885 \let\gl@sifplural\@firstoftwo
4886 \let\gl@scapscase\@firstofthree

```



```

4887 \let\glsinsert\@empty
4888 \def\glscustomtext{\glsxtrinlinefullplformat{#2}{#3}}%
4889 \glsxtrsetupfulldefs
4890 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4891 }%
4892 \glspostlinkhook
4893 }

```

\Glsxtrfullpl Plural full form (first letter uppercase).

```

4894 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
4895 \newcommand*\ns@Glsxtrfullpl[2][]{%
4896 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
4897 {\@Glsxtr@fullpl{#1}{#2}[]}%
4898 }

```

\@Glsxtr@fullpl Low-level macro:

```

4899 \def\@Glsxtr@fullpl#1#2[#3]{%
4900 \glsdoifexists{#2}%
4901 {%
4902 \glssetabbrvfmt{\glscategory{#2}}%
4903 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4904 \let\glsifplural\@firstoftwo
4905 \let\glscapscase\@secondofthree
4906 \let\glsinsert\@empty
4907 \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
4908 \glsxtrsetupfulldefs
4909 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4910 }%
4911 \glspostlinkhook
4912 }

```

\GLSxtrfullpl Plural full form (all upper case).

```

4913 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\@ns@GLSxtrfullpl}
4914 \newcommand*\ns@GLSxtrfullpl[2][]{%
4915 \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}%
4916 {\@GLSxtr@fullpl{#1}{#2}[]}%
4917 }

```

\@GLSxtr@fullpl Low-level macro:

```

4918 \def\@GLSxtr@fullpl#1#2[#3]{%
4919 \glsdoifexists{#2}%
4920 {%
4921 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4922 \let\glsifplural\@firstoftwo
4923 \let\glscapscase\@thirdofthree
4924 \let\glsinsert\@empty
4925 \def\glscustomtext{%
4926 \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
4927 \glsxtrsetupfulldefs

```

```

4928   \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4929   }%
4930   \glspostlinkhook
4931 }

```

The short and long forms work in a similar way to acronyms.

`\glsxtrshort`

```

4932 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4933 \newcommand*{\ns@glsxtrshort}[2] [] {%
4934   \new@ifnextchar[{\@glsxtrshort{#1}{#2}}{\@glsxtrshort{#1}{#2} []}%
4935 }

```

Read in the final optional argument:

```

4936 \def\@glsxtrshort#1#2[#3] {%
4937   \glsdoifexists{#2}%
4938   {%

```

Need to make sure `\glsabbrvfont` is set correctly.

```

4939   \glssetabbrvfmt{\glscategory{#2}}%
4940   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4941   \let\glxtrifwasfirstuse\@secondoftwo
4942   \let\glsifplural\@secondoftwo
4943   \let\glscapscase\@firstofthree
4944   \let\glsinsert\@empty
4945   \def\glscustomtext{%
4946     \glsabbrvfont{\glsaccessshort{#2}\ifglxtrininsertinside#3\fi}%
4947     \ifglxtrininsertinside\else#3\fi
4948   }%
4949   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4950   }%
4951   \glspostlinkhook
4952 }

```

`\Glsxtrshort`

```

4953 \newrobustcmd*{\Glsxtrshort}{\@gls@hyp@opt\ns@Glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4954 \newcommand*{\ns@Glsxtrshort}[2] [] {%
4955   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} []}%
4956 }

```

Read in the final optional argument:

```

4957 \def\@Glsxtrshort#1#2[#3] {%
4958   \glsdoifexists{#2}%
4959   {%
4960     \glssetabbrvfmt{\glscategory{#2}}%
4961     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4962     \let\glxtrifwasfirstuse\@secondoftwo

```

```

4963 \let\glsifplural\@secondoftwo
4964 \let\glscapscase\@secondofthree
4965 \let\glsinsert\@empty
4966 \def\glscustomtext{%
4967   \glsabbrvfont{\Glsaccessshort{#2}\ifglxtrinsertinside#3\fi}%
4968   \ifglxtrinsertinside\else#3\fi
4969 }%
4970 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4971 }%
4972 \glspostlinkhook
4973 }

```

\GLSxtrshort

```

4974 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
4975 \newcommand*{\ns@GLSxtrshort}[2][]{%
4976   \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2}[]}%
4977 }

    Read in the final optional argument:
4978 \def\@GLSxtrshort#1#2[#3]{%
4979   \glsdoifexists{#2}%
4980   {%
4981     \glssetabbrvfmt{\glscategory{#2}}%
4982     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4983     \let\glsxtrifwasfirstuse\@secondoftwo
4984     \let\glsifplural\@secondoftwo
4985     \let\glscapscase\@thirdofthree
4986     \let\glsinsert\@empty
4987     \def\glscustomtext{%
4988       \mfirstucMakeUppercase
4989       {\glsabbrvfont{\Glsaccessshort{#2}\ifglxtrinsertinside#3\fi}%
4990       \ifglxtrinsertinside\else#3\fi
4991     }%
4992   }%
4993   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4994 }%
4995 \glspostlinkhook
4996 }

```

\glsxtrlong

```

4997 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
4998 \newcommand*{\ns@glsxtrlong}[2][]{%
4999   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2}[]}%
5000 }

    Read in the final optional argument:
5001 \def\@glsxtrlong#1#2[#3]{%

```

```

5002 \glsdoifexists{#2}%
5003 {%
5004   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5005   \let\glxtrifwasfirstuse\@secondoftwo
5006   \let\gl@ifplural\@secondoftwo
5007   \let\glscapscase\@firstofthree
5008   \let\glinsert\@empty
5009   \def\glscustomtext{%
5010     \glslongfont{\gl@saccesslong{#2}\ifglxtrininsertinside#3\fi}%
5011     \ifglxtrininsertinside\else#3\fi
5012   }%
5013   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5014 }%
5015 \glspostlinkhook
5016 }

```

\Glsxtrlong

```

5017 \newrobustcmd*{\Glsxtrlong}{\@gl@hyp@opt\ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
5018 \newcommand*{\ns@Glsxtrlong}[2] [] {%
5019   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
5020 }
    Read in the final optional argument:
5021 \def\@Glsxtrlong#1#2[#3] {%
5022   \glsdoifexists{#2}%
5023   {%
5024     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5025     \let\glxtrifwasfirstuse\@secondoftwo
5026     \let\gl@ifplural\@secondoftwo
5027     \let\glscapscase\@secondofthree
5028     \let\glinsert\@empty
5029     \def\glscustomtext{%
5030       \glslongfont{\Glsaccesslong{#2}\ifglxtrininsertinside#3\fi}%
5031       \ifglxtrininsertinside\else#3\fi
5032     }%
5033     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5034   }%
5035   \glspostlinkhook
5036 }

```

\GLSxtrlong

```

5037 \newrobustcmd*{\GLSxtrlong}{\@gl@hyp@opt\ns@GLSxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
5038 \newcommand*{\ns@GLSxtrlong}[2] [] {%
5039   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
5040 }

```

Read in the final optional argument:

```

5041 \def\@GLSxtrlong#1#2[#3]{%
5042   \glsdoifexists{#2}%
5043   {%
5044     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5045     \let\glxtrifwasfirstuse\@secondoftwo
5046     \let\glsifplural\@secondoftwo
5047     \let\glscapscase\@thirdofthree
5048     \let\glsinsert\@empty
5049     \def\glscustomtext{%
5050       \mfirstucMakeUppercase
5051       {\glslongfont{\glsaccesslong{#2}\ifglxtrininsertinside#3\fi}%
5052       \ifglxtrininsertinside\else#3\fi
5053     }%
5054   }%
5055   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5056 }%
5057 \glspostlinkhook
5058 }

```

Plural short forms:

\glxtrshortpl

```

5059 \newrobustcmd*{\glxtrshortpl}{\@gl@hyp@opt\@ns@glxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

5060 \newcommand*{\ns@glxtrshortpl}[2][ ]{%
5061   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2}[]}%
5062 }

```

Read in the final optional argument:

```

5063 \def\@glxtrshortpl#1#2[#3]{%
5064   \glsdoifexists{#2}%
5065   {%
5066     \glsetabbrvfmt{\glscategory{#2}}%
5067     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5068     \let\glxtrifwasfirstuse\@secondoftwo
5069     \let\glsifplural\@firstoftwo
5070     \let\glscapscase\@firstofthree
5071     \let\glsinsert\@empty
5072     \def\glscustomtext{%
5073       \glsabbrvfont{\glsaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
5074       \ifglxtrininsertinside\else#3\fi
5075     }%
5076     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5077   }%
5078   \glspostlinkhook
5079 }

```

\Glsxtrshortpl

```

5080 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\@ns@Glsxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```
5081 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
5082   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
5083 }
```

Read in the final optional argument:

```
5084 \def\@Glsxtrshortpl#1#2[#3] {%
5085   \glsdoifexists{#2}%
5086   {%
5087     \glssetabbrvfmt{\glscategory{#2}}%
5088     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5089     \let\glsxtrifwasfirstuse\@secondoftwo
5090     \let\glsifplural\@firstoftwo
5091     \let\glsapscase\@secondofthree
5092     \let\glsinsert\@empty
5093     \def\glscustomtext{%
5094       \glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
5095       \ifglsxtrininsertinside\else#3\fi
5096     }%
5097     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5098   }%
5099   \glspostlinkhook
5100 }
```

\GLSxtrshortpl

```
5101 \newrobustcmd*{\GLSxtrshortpl}{\@gls@hyp@opt\ns@GLSxtrshortpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
5102 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
5103   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
5104 }
```

Read in the final optional argument:

```
5105 \def\@GLSxtrshortpl#1#2[#3] {%
5106   \glsdoifexists{#2}%
5107   {%
5108     \glssetabbrvfmt{\glscategory{#2}}%
5109     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5110     \let\glsxtrifwasfirstuse\@secondoftwo
5111     \let\glsifplural\@firstoftwo
5112     \let\glsapscase\@thirdofthree
5113     \let\glsinsert\@empty
5114     \def\glscustomtext{%
5115       \mfirstucMakeUppercase
5116       {\glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
5117       \ifglsxtrininsertinside\else#3\fi
5118     }%
5119   }%
5120   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5121   }%
```

```

5122 \glspostlinkhook
5123 }

```

Plural long forms:

\glsxtrlongpl

```

5124 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\@ns@glxtrlongpl}

Define the un-starred form. Need to determine if there is a final optional argument
5125 \newcommand*{\ns@glxtrlongpl}[2][\%
5126 \new@ifnextchar[\@glxtrlongpl{#1}{#2}]{\@glxtrlongpl{#1}{#2}[]}%
5127 }

Read in the final optional argument:
5128 \def\@glxtrlongpl#1#2[#3]{%
5129 \glstoifexists{#2}%
5130 {%
5131 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5132 \let\glxtrifwasfirstuse\@secondoftwo
5133 \let\gl@sifplural\@firstoftwo
5134 \let\glscapscase\@firstofthree
5135 \let\gl$insert\@empty
5136 \def\glscustomtext{%
5137 \glslongfont{\glssaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
5138 \ifglxtrininsertinside\else#3\fi
5139 }%
5140 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5141 }%
5142 \glspostlinkhook
5143 }

```

\Glsxtrlongpl

```

5144 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}

Define the un-starred form. Need to determine if there is a final optional argument
5145 \newcommand*{\ns@Glsxtrlongpl}[2][\%
5146 \new@ifnextchar[\@Glsxtrlongpl{#1}{#2}]{\@Glsxtrlongpl{#1}{#2}[]}%
5147 }

Read in the final optional argument:
5148 \def\@Glsxtrlongpl#1#2[#3]{%
5149 \glstoifexists{#2}%
5150 {%
5151 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5152 \let\glxtrifwasfirstuse\@secondoftwo
5153 \let\gl@sifplural\@firstoftwo
5154 \let\glscapscase\@secondofthree
5155 \let\gl$insert\@empty
5156 \def\glscustomtext{%
5157 \glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
5158 \ifglxtrininsertinside\else#3\fi

```

```

5159 }%
5160 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5161 }%
5162 \glspostlinkhook
5163 }

```

`\GLSxtrlongpl`

```

5164 \newrobustcmd*{\GLSxtrlongpl}{\@gls@hyp@opt\@ns@GLSxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
5165 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
5166   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
5167 }

    Read in the final optional argument:
5168 \def\@GLSxtrlongpl#1#2[#3]{%
5169   \glsdoifexists{#2}%
5170   {%
5171     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5172     \let\glstrifwasfirstuse\@secondoftwo
5173     \let\glsifplural\@firstoftwo
5174     \let\glscapscase\@thirdofthree
5175     \let\glsinsert\@empty
5176     \def\glscustomtext{%
5177       \mfirstucMakeUppercase
5178       {\glsfont{\glsaccesslongpl{#2}\ifglstrinsertinside#3\fi}%
5179       \ifglstrinsertinside\else#3\fi
5180     }%
5181   }%
5182   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5183 }%
5184 \glspostlinkhook
5185 }

```

`\glssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

5186 \newcommand*{\glssetabbrvfmt}[1]{%
5187   \ifcsdef{@glsabbrv@current@#1}%
5188   {\glsxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
5189   {\glsxtr@applyabbrvfmt{@glsabbrv@current@abbreviation}}%
5190 }

```

`sxtrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

5191 \newcommand*{\glsxtrgenabbrvfmt}{%
5192   \ifdefempty\glscustomtext
5193   {%
5194     \ifglssused\glslabel
5195     {%

```


Subsequent use:

```
5196      \glsifplural
5197      {%
```

Subsequent plural form:

```
5198      \glscapscase
5199      {%
```

Subsequent plural form, don't adjust case:

```
5200      \glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert
5201      }%
5202      {%
```

Subsequent plural form, make first letter upper case:

```
5203      \glsabbrvfont{\Glsaccessshortpl{\glslabel}}\glsinsert
5204      }%
5205      {%
```

Subsequent plural form, all caps:

```
5206      \mfirstucMakeUppercase
5207      {\glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert}%
5208      }%
5209      }%
5210      {%
```

Subsequent singular form

```
5211      \glscapscase
5212      {%
```

Subsequent singular form, don't adjust case:

```
5213      \glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert
5214      }%
5215      {%
```

Subsequent singular form, make first letter upper case:

```
5216      \glsabbrvfont{\Glsaccessshort{\glslabel}}\glsinsert
5217      }%
5218      {%
```

Subsequent singular form, all caps:

```
5219      \mfirstucMakeUppercase
5220      {\glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert}%
5221      }%
5222      }%
5223      }%
5224      {%
```

First use:

```
5225      \glsifplural
5226      {%
```

First use plural form:

```
5227      \glscapscase
5228      {%
```

First use plural form, don't adjust case:

```
5229      \glstrfullplformat{\glslabel}{\glsinsert}%
5230      }%
5231      {%
```

First use plural form, make first letter upper case:

```
5232      \Glsstrfullplformat{\glslabel}{\glsinsert}%
5233      }%
5234      {%
```

First use plural form, all caps:

```
5235      \mfirstucMakeUppercase
5236      {\glstrfullplformat{\glslabel}{\glsinsert}}%
5237      }%
5238      }%
5239      {%
```

First use singular form

```
5240      \glscapscase
5241      {%
```

First use singular form, don't adjust case:

```
5242      \glstrfullformat{\glslabel}{\glsinsert}%
5243      }%
5244      {%
```

First use singular form, make first letter upper case:

```
5245      \Glsstrfullformat{\glslabel}{\glsinsert}%
5246      }%
5247      {%
```

First use singular form, all caps:

```
5248      \mfirstucMakeUppercase
5249      {\glstrfullformat{\glslabel}{\glsinsert}}%
5250      }%
5251      }%
5252      }%
5253      }%
5254      {%
```

User supplied text.

```
5255      \glscustomtext
5256      }%
5257 }
```

1.6.1 Abbreviation Styles Setup

abbreviationstyle

```
5258 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
5259   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
5260   {%
```

```

5261 \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
5262 }%
5263 {%

```

Have abbreviations already been defined for this category?

```

5264 \ifcsstring{@glsabbrv@current@#1}{#2}%
5265 {%

```

Style already set.

```

5266 }%
5267 {%
5268 \def\@glsxtr@dostylewarn{%
5269 \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
5270 {%
5271 \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
5272 style has been switched \MessageBreak
5273 for category ‘#1’, \MessageBreak
5274 but there have already been entries \MessageBreak
5275 defined for this category. Unwanted \MessageBreak
5276 side-effects may result}}%
5277 \@endfortrue
5278 }%
5279 \@glsxtr@dostylewarn

```

Set up the style for the given category.

```

5280 \csdef{@glsabbrv@current@#1}{#2}%
5281 \glsxtr@applyabbrvstyle{#2}%
5282 }%
5283 }%
5284 }

```

applyabbrvstyle Apply the abbreviation style without existence check.

```

5285 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
5286 \csuse{@glsabbrv@dispstyle@setup@#1}%
5287 \csuse{@glsabbrv@dispstyle@fmts@#1}%
5288 }

```

r@applyabbrvfmt Only apply the style formats.

```

5289 \newcommand*{\glsxtr@applyabbrvfmt}[1]{%
5290 \csuse{@glsabbrv@dispstyle@fmts@#1}%
5291 }

```

abbreviationstyle This is different from `\newacronymstyle`. The first argument is the label, the second argument sets the information required when defining the new abbreviation and the third argument sets the commands used to display the full format.

```

5292 \newcommand*{\newabbreviationstyle}[3]{%
5293 \ifcsdef{@glsabbrv@dispstyle@setup@#1}
5294 {%
5295 \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
5296 defined}{}%

```

```

5297 }%
5298 {%
5299 \csdef{@glsabbrv@dispstyle@setup@#1}{%
    Initialise hook to do nothing. The style may change this.
5300 \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
5301 #2}%
5302 \csdef{@glsabbrv@dispstyle@fmts@#1}{%
    Assume in-line form is the same as first use. The style may change this.
5303 \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
5304 \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
5305 \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
5306 \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
5307 #3}%
5308 }%
5309 }

```

breivationstyle

```

5310 \newcommand*{\renewabbreviationstyle}[3]{%
5311 \ifcsundef{@glsabbrv@dispstyle@setup@#1}
5312 {%
5313 \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
5314 }%
5315 {%
5316 \csdef{@glsabbrv@dispstyle@setup@#1}{%
    Initialise hook to do nothing. The style may change this.
5317 \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
5318 #2}%
5319 \csdef{@glsabbrv@dispstyle@fmts@#1}{%
    Assume in-line form is the same as first use. The style may change this.
5320 \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
5321 \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
5322 \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
5323 \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
5324 #3}%
5325 }%
5326 }

```

breivationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style’s name.

```

5327 \newcommand*{\letabbreviationstyle}[2]{%
5328 \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
5329 \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
5330 }

```

ecated@abbrstyle `\glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

5331 \newcommand*{\@GlsXtrDeprecatedAbbrStyle}[2]{%
5332   \csdef{@Glsabbrv@dispstyle@setup@#1}{%
5333     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
5334     \csuse{@Glsabbrv@dispstyle@setup@#2}%
5335   }%
5336   \csletcs{@Glsabbrv@dispstyle@fmts@#1}{@Glsabbrv@dispstyle@fmts@#2}%
5337 }
```

`\GlsXtrWarnDeprecatedAbbrStyle` Generate warning for deprecated style use.

```

5338 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
5339   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
5340     use ‘#2’ instead}%
5341 }
```

`\GlsXtrUseAbbrStyleSetup`

```

5342 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
5343   \ifcsundef{@Glsabbrv@dispstyle@setup@#1}%
5344   {%
5345     \PackageError{glossaries-extra}%
5346       {Unknown abbreviation style definitions ‘#1’}{}%
5347   }%
5348   {%
5349     \csname @Glsabbrv@dispstyle@setup@#1\endcsname
5350   }%
5351 }
```

`\GlsXtrUseAbbrStyleFmts`

```

5352 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
5353   \ifcsundef{@Glsabbrv@dispstyle@fmts@#1}%
5354   {%
5355     \PackageError{glossaries-extra}%
5356       {Unknown abbreviation style formats ‘#1’}{}%
5357   }%
5358   {%
5359     \csname @Glsabbrv@dispstyle@fmts@#1\endcsname
5360   }%
5361 }
```

1.6.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn't set to “true”. If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glsxtrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

xtrinsertinside Switch to determine if the insert text should be inside or outside the font changing command.

The default is outside.

```
5362 \newif\ifglxtrinsertinside
5363 \glxtrinsertinsidefalse
```

long-short

```
5364 \newabbreviationstyle{long-short}%
5365 {%
5366   \renewcommand*{\CustomAbbreviationFields}{%
5367     name={\protect\glssabrvfont{\the\glsshorttok}},
5368     sort={\the\glsshorttok},
5369     first={\protect\glsfirstlongfont{\the\glslongtok}%
5370       \protect\glxtrfullsep{\the\glslabeltok}%
5371       (\protect\glsfirstabrvfont{\the\glsshorttok})},%
5372     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
5373       \protect\glxtrfullsep{\the\glslabeltok}%
5374       (\protect\glsfirstabrvfont{\the\glsshortpltok})},%
5375     plural={\protect\glssabrvfont{\the\glsshortpltok}},%
5376     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
5377 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5378   \glshasattribute{\the\glslabeltok}{regular}%
5379   {%
5380     \glissetattribute{\the\glslabeltok}{regular}{false}%
5381   }%
5382 }%
5383 }%
5384 }%
5385 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5386 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
5387 \renewcommand*{\glssabrvfont}[1]{\glssabrvdefaultfont{##1}}%
5388 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvdefaultfont{##1}}%
5389 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5390 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
5391 \renewcommand*{\glxtrfullformat}[2]{%
5392   \glsfirstlongfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
5393   \ifglxtrinsertinside\else##2\fi
5394   \glxtrfullsep{##1}%
5395   (\glsfirstabrvfont{\glssaccessshort{##1}})%
5396 }%
5397 \renewcommand*{\glxtrfullplformat}[2]{%
5398   \glsfirstlongfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5399   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5400   (\glsfirstabrvfont{\glssaccessshortpl{##1}})%
5401 }%
```

```

5402 \renewcommand*{\Glsxtrfullformat}[2]{%
5403   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
5404   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5405   (\glsfirstabbrvfont{\glsaccessshort{##1}})%
5406 }%
5407 \renewcommand*{\Glsxtrfullplformat}[2]{%
5408   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5409   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5410   (\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
5411 }%
5412 }

```

Set this as the default style for general abbreviations:

```

5413 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

5414 \newcommand*{\glxtrlongshortdescsort}{\the\glslongtok\space(\the\glsshorttok)}

```

long-short-desc User supplies description. The long form is included in the name.

```

5415 \newabbreviationstyle{long-short-desc}%
5416 {%
5417   \renewcommand*{\CustomAbbreviationFields}{%
5418     name={\protect\glxtrfullformat{\the\glslabeltok}{}},
5419     sort={\glxtrlongshortdescsort},%
5420     first={\protect\glsfirstlongfont{\the\glslongtok}%
5421       \protect\glxtrfullsep{\the\glslabeltok}%
5422       (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
5423     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
5424       \protect\glxtrfullsep{\the\glslabeltok}%
5425       (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%

```

The text key should only have the short form.

```

5426   text={\protect\glsabbrvfont{\the\glsshorttok}},%
5427   plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
5428 }%

```

Unset the regular attribute if it has been set.

```

5429 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5430   \glshasattribute{\the\glslabeltok}{regular}%
5431   {%
5432     \glissetattribute{\the\glslabeltok}{regular}{false}%
5433   }%
5434   {}%
5435 }%
5436 }%
5437 {%
5438   \GlsXtrUseAbbrStyleFmts{long-short}%
5439 }

```

short-long Short form followed by long form in parenthesis on first use.

```
5440 \newabbreviationstyle{short-long}%
5441 {%
5442   \renewcommand*{\CustomAbbreviationFields}{%
5443     name={\protect\glsabbrvfont{\the\glsshorttok}},
5444     sort={\the\glsshorttok},
5445     description={\the\glslongtok},%
5446     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
5447       \protect\glsxtrfullsep{\the\glslabeltok}%
5448       (\protect\glsfirstlongfont{\the\glslongtok})},%
5449     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
5450       \protect\glsxtrfullsep{\the\glslabeltok}%
5451       (\protect\glsfirstlongfont{\the\glslongpltok})},%
5452     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
5453   }
```

Unset the regular attribute if it has been set.

```
5453   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5454     \glshasattribute{\the\glslabeltok}{regular}%
5455     {%
5456       \glssetattribute{\the\glslabeltok}{regular}{false}%
5457     }%
5458   }%
5459 }%
5460 }%
5461 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5462 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
5463 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5464 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5465 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5466 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
5467 \renewcommand*{\glsxtrfullformat}[2]{%
5468   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5469   \ifglsxtrininsertinside\else##2\fi
5470   \glsxtrfullsep{##1}%
5471   (\glsfirstlongfont{\glsaccesslong{##1}})%
5472 }%
5473 \renewcommand*{\glsxtrfullplformat}[2]{%
5474   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5475   \ifglsxtrininsertinside\else##2\fi
5476   \glsxtrfullsep{##1}%
5477   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
5478 }%
5479 \renewcommand*{\Glsxtrfullformat}[2]{%
5480   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5481   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5482   (\glsfirstlongfont{\Glsaccesslong{##1}})%
5483 }
```



```

5483 }%
5484 \renewcommand*{\Glsxtrfullplformat}[2]{%
5485   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5486   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5487   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
5488 }%
5489 }

```

short-long-desc User supplies description. The long form is included in the name.

```

5490 \newabbreviationstyle{short-long-desc}%
5491 {%
5492   \renewcommand*{\CustomAbbreviationFields}{%
5493     name={\protect\glsxtrfullformat{\the\glslabeltok}{}},
5494     sort={\the\glsshorttok},%
5495     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
5496       \protect\glsxtrfullsep{\the\glslabeltok}%
5497       (\protect\glsfirstlongfont{\the\glslongtok})},%
5498     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
5499       \protect\glsxtrfullsep{\the\glslabeltok}%
5500       (\protect\glsfirstlongfont{\the\glslongpltok})},%
5501     text={\protect\glsabbrvfont{\the\glsshorttok}},%
5502     plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
5503   }%

```

Unset the regular attribute if it has been set.

```

5504 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5505   \glshasattribute{\the\glslabeltok}{regular}%
5506   {%
5507     \glssetattribute{\the\glslabeltok}{regular}{false}%
5508   }%
5509   {}%
5510 }%
5511 }%
5512 {%
5513   \GlsXtrUseAbbrStyleFmts{short-long}%
5514 }

```

ongfootnotefont Only used by the “footnote” styles.

```

5515 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{##1}}%

```

ongfootnotefont Only used by the “footnote” styles.

```

5516 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{##1}}%

```

xtrabbrvfootnote `\glsxtrabbrvfootnote{<label>}{<long>}`

Command used by footnote abbreviation styles. The default definition ignores the first argument. The second argument *<long>* includes the font changing command and may be the singular or plural form, depending on the command that was used (for example, `\gls` or `\glspl`).

```
5517 \newcommand*{\glxtrabbrvfootnote}[2]{\footnote{#2}}
```

footnote Short form followed by long form in footnote on first use.

```
5518 \newabbreviationstyle{footnote}%
5519 {%
5520   \renewcommand*{\CustomAbbreviationFields}{%
5521     name={\protect\glsabbrvfont{\the\glsshorttok}},
5522     sort={\the\glsshorttok},
5523     description={\the\glslongtok},%
5524     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
5525       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
5526       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
5527     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
5528       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
5529       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
5530     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
5531 }
```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
5531 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5532   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
5533   \glshasattribute{\the\glslabeltok}{regular}%
5534   {%
5535     \glssetattribute{\the\glslabeltok}{regular}{false}%
5536   }%
5537   {}%
5538 }%
5539 }%
5540 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5541 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
5542 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5543 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5544 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
5545 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
5546 \renewcommand*{\glxtrfullformat}[2]{%
5547   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
5548   \ifglxtrininsertinside\else##2\fi
5549   \protect\glxtrabbrvfootnote{##1}%
5550   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
5551 }%
5552 \renewcommand*{\glxtrfullplformat}[2]{%
```

```

5553 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5554 \ifglxtrinsertinside\else##2\fi
5555 \protect\glxtrabbrvfootnote{##1}%
5556 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
5557 }%
5558 \renewcommand*{\Glsxtrfullformat}[2]{%
5559 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5560 \ifglxtrinsertinside\else##2\fi
5561 \protect\glxtrabbrvfootnote{##1}%
5562 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
5563 }%
5564 \renewcommand*{\Glsxtrfullplformat}[2]{%
5565 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5566 \ifglxtrinsertinside\else##2\fi
5567 \protect\glxtrabbrvfootnote{##1}%
5568 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
5569 }%

```

The first use full form and the inline full form use the short (long) style.

```

5570 \renewcommand*{\glsxtrinlinefullformat}[2]{%
5571 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5572 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5573 (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
5574 }%
5575 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
5576 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5577 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5578 (\glsfirstlongfootnotefont{\glsaccesslongpl{##1}})%
5579 }%
5580 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
5581 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5582 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5583 (\glsfirstlongfootnotefont{\Glsaccesslong{##1}})%
5584 }%
5585 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
5586 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5587 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5588 (\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}})%
5589 }%
5590 }

```

short-footnote

```
5591 \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.

```

5592 \newabbreviationstyle{postfootnote}%
5593 {}

```

```

5594 \renewcommand*{\CustomAbbreviationFields}{%
5595   name={\protect\glsabbrvfont{\the\glsshorttok}},
5596   sort={\the\glsshorttok},
5597   description={\the\glslongtok},%
5598   first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
5599   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
5600   plural={\protect\glsabbvfont{\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.

```

5601 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5602   \csdef{glsxtrpostlink\glscategorylabel}{%
5603     \glsxtrifwasfirstuse
5604     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

5605       \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}}%
5606       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
5607     }%
5608   }%
5609 }%
5610 \glsattribute{\the\glslabeltok}{regular}%
5611 {%
5612   \glssetattribute{\the\glslabeltok}{regular}{false}%
5613 }%
5614 {}%
5615 }%

```

The footnote needs to be suppressed in the inline form, so `\glsxtrfull` must set the first use switch off.

```

5616 \renewcommand*{\glsxtrsetupfulldefs}{%
5617   \let\glsxtrifwasfirstuse\@secondoftwo
5618 }%
5619 }%
5620 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5621 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
5622 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5623 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5624 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
5625 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

5626 \renewcommand*{\glsxtrfullformat}[2]{%
5627   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5628   \ifglsxtrininsertinside\else##2\fi
5629 }%
5630 \renewcommand*{\glsxtrfullplformat}[2]{%
5631   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%

```

```

5632 \ifglxtrinsertinside\else##2\fi
5633 }%
5634 \renewcommand*{\Glsxtrfullformat}[2]{%
5635 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5636 \ifglxtrinsertinside\else##2\fi
5637 }%
5638 \renewcommand*{\Glsxtrfullplformat}[2]{%
5639 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5640 \ifglxtrinsertinside\else##2\fi
5641 }%

```

The first use full form and the inline full form use the short (long) style.

```

5642 \renewcommand*{\glxtrinlinefullformat}[2]{%
5643 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5644 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5645 (\glsfirstlongfootnotefont{\Glsaccesslong{##1}})%
5646 }%
5647 \renewcommand*{\glxtrinlinefullplformat}[2]{%
5648 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5649 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5650 (\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}})%
5651 }%
5652 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
5653 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5654 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5655 (\glsfirstlongfootnotefont{\Glsaccesslong{##1}})%
5656 }%
5657 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
5658 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5659 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5660 (\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}})%
5661 }%
5662 }

```

rt-postfootnote

```

5663 \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.

```

5664 \newabbreviationstyle{short}%
5665 {%
5666 \renewcommand*{\CustomAbbreviationFields}{%
5667 name={\protect\glsabbrvfont{\the\glsshorttok}},
5668 sort={\the\glsshorttok},
5669 first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
5670 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
5671 text={\protect\glsabbrvfont{\the\glsshorttok}},

```

```

5672 plural={\protect\glsabbrvfont{\the\glsshortpltok}},
5673 description={\the\glslongtok}}%
5674 \renewcommand*\GlsXtrPostNewAbbreviation}{%
5675   \glssetattribute{\the\glslabeltok}{regular}{true}}%
5676 }%
5677 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5678 \renewcommand*\abbrvpluralsuffix{\glsxtrabbrvpluralsuffix}%
5679 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
5680 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
5681 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
5682 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

5683 \renewcommand*\glsxtrinlinefullformat[2]{%
5684   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
5685   \ifglsxtrininsertinside##2\fi}%
5686   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5687   (\glsfirstlongfont{\glsaccesslong{##1}})%
5688 }%
5689 \renewcommand*\glsxtrinlinefullplformat[2]{%
5690   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
5691   \ifglsxtrininsertinside##2\fi}%
5692   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5693   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
5694 }%
5695 \renewcommand*\Glsxtrinlinefullformat[2]{%
5696   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
5697   \ifglsxtrininsertinside##2\fi}%
5698   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5699   (\glsfirstlongfont{\Glsaccesslong{##1}})%
5700 }%
5701 \renewcommand*\Glsxtrinlinefullplformat[2]{%
5702   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
5703   \ifglsxtrininsertinside##2\fi}%
5704   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5705   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
5706 }%

```

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.

```

5707 \renewcommand*\glsxtrfullformat[2]{%
5708   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
5709   \ifglsxtrininsertinside\else##2\fi
5710 }%
5711 \renewcommand*\glsxtrfullplformat[2]{%
5712   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
5713   \ifglsxtrininsertinside\else##2\fi
5714 }%
5715 \renewcommand*\Glsxtrfullformat[2]{%

```

```

5716 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5717 \ifglxtrinsertinside\else##2\fi
5718 }%
5719 \renewcommand*{\Glsxtrfullplformat}[2]{%
5720 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5721 \ifglxtrinsertinside\else##2\fi
5722 }%
5723 }

```

Set this as the default style for acronyms:

```
5724 \setabbreviationstyle[acronym]{short}
```

short-nolong

```
5725 \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.

```

5726 \newabbreviationstyle{short-desc}%
5727 {%
5728 \renewcommand*{\CustomAbbreviationFields}{%
5729 name={\protect\glsxtrinlinefullformat{\the\glslabeltok}{}},
5730 sort={\the\glsshorttok},
5731 first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
5732 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
5733 text={\protect\glsabbrvfont{\the\glsshorttok}},
5734 plural={\protect\glsabbrvfont{\the\glsshortpltok}},
5735 description={\the\glslongtok}}%
5736 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5737 \glssetattribute{\the\glslabeltok}{regular}{true}}%
5738 }%
5739 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5740 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
5741 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5742 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5743 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5744 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

5745 \renewcommand*{\glsxtrinlinefullformat}[2]{%
5746 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5747 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5748 (\glsfirstlongfont{\glsaccesslong{##1}})%
5749 }%
5750 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
5751 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5752 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
5753 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
5754 }%

```

```

5755 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
5756   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5757   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5758   (\glsfirstlongfont{\Glsaccesslong{##1}})%
5759 }%
5760 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
5761   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5762   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5763   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
5764 }%

```

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.

```

5765 \renewcommand*{\glxtrfullformat}[2]{%
5766   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5767   \ifglxtrinsertinside\else##2\fi
5768 }%
5769 \renewcommand*{\glxtrfullplformat}[2]{%
5770   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5771   \ifglxtrinsertinside\else##2\fi
5772 }%
5773 \renewcommand*{\Glsxtrfullformat}[2]{%
5774   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5775   \ifglxtrinsertinside\else##2\fi
5776 }%
5777 \renewcommand*{\Glsxtrfullplformat}[2]{%
5778   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5779   \ifglxtrinsertinside\else##2\fi
5780 }%
5781 }

```

ort-nolong-desc

```
5782 \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.

```

5783 \newabbreviationstyle{long-desc}%
5784 {%
5785   \renewcommand*{\CustomAbbreviationFields}{%
5786     name={\protect\protect\glsfirstlongfont{\the\glslongtok}},
5787     sort={\the\glslongtok},
5788     first={\protect\glsfirstlongfont{\the\glslongtok}},
5789     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5790     text={\the\glslongtok},
5791     plural={\the\glslongpltok}%
5792   }%
5793   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5794     \glssetattribute{\the\glslabeltok}{regular}{true}}%

```


5795 }%
 5796 {%

In case the user wants to mix and match font styles, these are redefined here.

```
5797 \renewcommand*\abbrvpluralsuffix{\glstrabbrvpluralsuffix}%
5798 \renewcommand*\glssabbrvfont[1]{\glssabbrvdefaultfont{##1}}%
5799 \renewcommand*\glssfirstabbrvfont[1]{\glssfirstabbrvdefaultfont{##1}}%
5800 \renewcommand*\glssfirstlongfont[1]{\glssfirstlongdefaultfont{##1}}%
5801 \renewcommand*\glsslongfont[1]{\glsslongdefaultfont{##1}}%
```

The inline full form displays the long format followed by the short form in parentheses.

```
5802 \renewcommand*\glssxtrinlinefullformat[2]{%
5803   \glssfirstlongfont{\glssaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
5804   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
5805   (\protect\glssfirstabbrvfont{\glssaccessshort{##1}})%
5806 }%
5807 \renewcommand*\glssxtrinlinefullplformat[2]{%
5808   \glssfirstlongfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
5809   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
5810   (\protect\glssfirstabbrvfont{\glssaccessshortpl{##1}})%
5811 }%
5812 \renewcommand*\Glsxtrinlinefullformat[2]{%
5813   \glssfirstlongfont{\Glsaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
5814   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
5815   (\protect\glssfirstabbrvfont{\Glsaccessshort{##1}})%
5816 }%
5817 \renewcommand*\Glsxtrinlinefullplformat[2]{%
5818   \glssfirstlongfont{\Glsaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
5819   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
5820   (\protect\glssfirstabbrvfont{\Glsaccessshortpl{##1}})%
5821 }%
```

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.

```
5822 \renewcommand*\glssxtrfullformat[2]{%
5823   \glssfirstlongfont{\glssaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
5824   \ifglssxtrininsertinside\else##2\fi
5825 }%
5826 \renewcommand*\glssxtrfullplformat[2]{%
5827   \glssfirstlongfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
5828   \ifglssxtrininsertinside\else##2\fi
5829 }%
5830 \renewcommand*\Glsxtrfullformat[2]{%
5831   \glssfirstlongfont{\Glsaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
5832   \ifglssxtrininsertinside\else##2\fi
5833 }%
5834 \renewcommand*\Glsxtrfullplformat[2]{%
5835   \glssfirstlongfont{\Glsaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
5836   \ifglssxtrininsertinside\else##2\fi
5837 }%
5838 }
```

ng-noshort-desc Provide a synonym that matches similar styles.

```
5839 \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.

```
5840 \newabbreviationstyle{long}%
5841 {%
5842   \renewcommand*{\CustomAbbreviationFields}{%
5843     name={\protect\glsabbrvfont{\the\glsshorttok}},
5844     sort={\the\glsshorttok},
5845     first={\protect\glsfirstlongfont{\the\glslongtok}},
5846     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5847     text={\the\glslongtok},
5848     plural={\the\glslongpltok},%
5849     description={\the\glslongtok}%
5850   }%
5851   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5852     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5853 }%
5854 {%
5855   \GlsXtrUseAbbrStyleFmts{long-desc}%
5856 }
```

long-noshort Provide a synonym that matches similar styles.

```
5857 \letabbreviationstyle{long-noshort}{long}
```

1.6.3 Predefined Styles (Small Capitals)

These styles use:

\glsxtrscfont

```
5858 \newcommand*{\glsxtrscfont}[1]{\textsc{#1}}
```

sxtrfirstscfont

```
5859 \newcommand*{\glsxtrfirstscfont}[1]{\glsxtrscfont{#1}}
```

and for the default short form suffix:

\glsxtrscsuffix

```
5860 \newcommand*{\glsxtrscsuffix}{\glsfontup{\glsxtrabbrvpluralsuffix}}
```

long-short-sc

```
5861 \newabbreviationstyle{long-short-sc}%
5862 {%
5863   \GlsXtrUseAbbrStyleSetup{long-short}%
5864 }%
5865 {%
```

Mostly as long-short style:

```
5866 \GlsXtrUseAbbrStyleFmts{long-short}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5867 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5868 \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%
5869 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%
5870 }
```

g-short-sc-desc

```
5871 \newabbreviationstyle{long-short-sc-desc}%
5872 {%
5873 \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5874 }%
5875 {%
```

Mostly as long-short-desc style:

```
5876 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5877 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5878 \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%
5879 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%
5880 }
```

Now the short (long) version

```
5881 \newabbreviationstyle{short-sc-long}%
5882 {%
5883 \GlsXtrUseAbbrStyleSetup{short-long}%
5884 }%
5885 {%
```

Mostly as short-long style:

```
5886 \GlsXtrUseAbbrStyleFmts{short-long}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5887 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5888 \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%
5889 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%
5890 }
```

As before but user provides description

```
5891 \newabbreviationstyle{short-sc-long-desc}%
5892 {%
5893 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5894 }%
5895 {%
```

Mostly as short-long-desc style:

```
5896 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5897 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5898 \renewcommand*{\glssabrvfont[1]}{\glxtrscfont{##1}}%
5899 \renewcommand*{\glssfirstabrvfont[1]}{\glxtrfirstscfont{##1}}%
5900 }
```

short-sc

```
5901 \newabbreviationstyle{short-sc}%
5902 {%
5903   \GlsXtrUseAbbrStyleSetup{short-nolong}%
5904 }%
5905 {%
```

Mostly as short style:

```
5906   \GlsXtrUseAbbrStyleFmts{short-nolong}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5907 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5908 \renewcommand*{\glssabrvfont[1]}{\glxtrscfont{##1}}%
5909 \renewcommand*{\glssfirstabrvfont[1]}{\glxtrfirstscfont{##1}}%
5910 }
```

short-sc-nolong

```
5911 \letabbreviationstyle{short-sc-nolong}{short-sc}
```

short-sc-desc

```
5912 \newabbreviationstyle{short-sc-desc}%
5913 {%
5914   \GlsXtrUseAbbrStyleSetup{short-desc}%
5915 }%
5916 {%
```

Mostly as short style:

```
5917   \GlsXtrUseAbbrStyleFmts{short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5918 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5919 \renewcommand*{\glssabrvfont[1]}{\glxtrscfont{##1}}%
5920 \renewcommand*{\glssfirstabrvfont[1]}{\glxtrfirstscfont{##1}}%
5921 }
```

-sc-nolong-desc

```
5922 \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 \glsshort.

```
5923 \newabbreviationstyle{long-noshort-sc}%
5924 {%
5925   \GlsXtrUseAbbrStyleSetup{long-noshort}%
5926 }%
5927 {%
```

Mostly as long style:

```
5928 \GlsXtrUseAbbrStyleFmts{long-noshort}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5929 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
5930 \renewcommand*{\glssabrvfont[1]{\glstrscfont{##1}}}%
5931 \renewcommand*{\glssfirstabrvfont[1]{\glstrfirstscfont{##1}}}%
5932 }
```

long-sc Backward compatibility:

```
5933 \@glstr@deprecated@abbrstyle{long-sc}{long-noshort-sc}
```

noshort-sc-desc The smallcaps font will only be used if the short form is explicitly invoked through commands like \glsshort.

```
5934 \newabbreviationstyle{long-noshort-sc-desc}%
5935 {%
5936 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5937 }%
5938 {%
```

Mostly as long style:

```
5939 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5940 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
5941 \renewcommand*{\glssabrvfont[1]{\glstrscfont{##1}}}%
5942 \renewcommand*{\glssfirstabrvfont[1]{\glstrfirstscfont{##1}}}%
5943 }
```

long-desc-sc Backward compatibility:

```
5944 \@glstr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}
```

ort-sc-footnote

```
5945 \newabbreviationstyle{short-sc-footnote}%
5946 {%
5947 \GlsXtrUseAbbrStyleSetup{short-footnote}%
5948 }%
5949 {%
```

Mostly as long style:

```
5950 \GlsXtrUseAbbrStyleFmts{short-footnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5951 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
5952 \renewcommand*{\glssabrvfont[1]{\glstrscfont{##1}}}%
5953 \renewcommand*{\glssfirstabrvfont[1]{\glstrfirstscfont{##1}}}%
5954 }
```

footnote-sc Backward compatibility:

```
5955 \@glstr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}
```

sc-postfootnote

```
5956 \newabbreviationstyle{short-sc-postfootnote}%  
5957 {%  
5958   \GlsXtrUseAbbrStyleSetup{short-postfootnote}%  
5959 }%  
5960 {%
```

Mostly as long style:

```
5961   \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5962   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%  
5963   \renewcommand*\glsabbrvfont[1]{\glstrscfont{##1}}%  
5964   \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstscfont{##1}}%  
5965 }
```

postfootnote-sc Backward compatibility:

```
5966 \@glstr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}
```

1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the `relese` package, which must be loaded by the user. These styles all use:

`\glstrsmfont`

```
5967 \newcommand*{\glstrsmfont}[1]{\textsmaller{#1}}
```

`glstrfirstsmfont`

```
5968 \newcommand*{\glstrfirstsmfont}[1]{\glstrsmfont{#1}}
```

and for the default short form suffix:

`\glstrsmsuffix`

```
5969 \newcommand*{\glstrsmsuffix}{\glstrabbrvpluralsuffix}
```

long-short-sm

```
5970 \newabbreviationstyle{long-short-sm}%  
5971 {%  
5972   \GlsXtrUseAbbrStyleSetup{long-short}%  
5973 }%  
5974 {%
```

Mostly as long-short style:

```
5975   \GlsXtrUseAbbrStyleFmts{long-short}%  
5976   \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%  
5977   \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}%  
5978   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrsmsuffix}%  
5979 }
```

g-short-sm-desc

```
5980 \newabbreviationstyle{long-short-sm-desc}%
5981 {%
5982   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5983 }%
5984 {%
```

Mostly as long-short-desc style:

```
5985   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
5986   \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
5987   \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}%
5988   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrsmsuffix}%
5989 }
```

short-sm-long Now the short (long) version

```
5990 \newabbreviationstyle{short-sm-long}%
5991 {%
5992   \GlsXtrUseAbbrStyleSetup{short-long}%
5993 }%
5994 {%
```

Mostly as short-long style:

```
5995   \GlsXtrUseAbbrStyleFmts{short-long}%
5996   \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
5997   \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}%
5998   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrsmsuffix}%
5999 }
```

rt-sm-long-desc As before but user provides description

```
6000 \newabbreviationstyle{short-sm-long-desc}%
6001 {%
6002   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
6003 }%
6004 {%
```

Mostly as short-long-desc style:

```
6005   \GlsXtrUseAbbrStyleFmts{short-long-desc}%
6006   \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
6007   \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}%
6008   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrsmsuffix}%
6009 }
```

short-sm

```
6010 \newabbreviationstyle{short-sm}%
6011 {%
6012   \GlsXtrUseAbbrStyleSetup{short-nolong}%
6013 }%
6014 {%
```

Mostly as short style:

```
6015 \GlsXtrUseAbbrStyleFmts{short-nolong}%
6016 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
6017 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
6018 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6019 }
```

short-sm-nolong

```
6020 \letabbreviationstyle{short-sm-nolong}{short-sm}
```

short-sm-desc

```
6021 \newabbreviationstyle{short-sm-desc}%
6022 {%
6023 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
6024 }%
6025 {%
```

Mostly as short style:

```
6026 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
6027 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
6028 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
6029 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6030 }
```

-sm-nolong-desc

```
6031 \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.

```
6032 \newabbreviationstyle{long-noshort-sm}%
6033 {%
6034 \GlsXtrUseAbbrStyleSetup{long-noshort}%
6035 }%
6036 {%
```

Mostly as long style:

```
6037 \GlsXtrUseAbbrStyleFmts{long-noshort}%
6038 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
6039 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
6040 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6041 }
```

long-sm Backward compatibility:

```
6042 \@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.

```
6043 \newabbreviationstyle{long-noshort-sm-desc}%
6044 {%
```



```

6045 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
6046 }%
6047 {%

```

Mostly as long style:

```

6048 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
6049 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
6050 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
6051 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6052 }

```

long-desc-sm Backward compatibility:

```

6053 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```

6054 \newabbreviationstyle{short-sm-footnote}%
6055 {%
6056 \GlsXtrUseAbbrStyleSetup{short-footnote}%
6057 }%
6058 {%

```

Mostly as long style:

```

6059 \GlsXtrUseAbbrStyleFmts{short-footnote}%
6060 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
6061 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
6062 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6063 }

```

footnote-sm Backward compatibility:

```

6064 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

sm-postfootnote

```

6065 \newabbreviationstyle{short-sm-postfootnote}%
6066 {%
6067 \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
6068 }%
6069 {%

```

Mostly as long style:

```

6070 \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
6071 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
6072 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
6073 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
6074 }

```

postfootnote-sm Backward compatibility:

```

6075 \@glxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```
6076 \newcommand*{\glsabbrvemfont}[1]{\emph{#1}}%
```

`\glsfirstabbrvemfont`

```
6077 \newcommand*{\glsfirstabbrvemfont}[1]{\glsabbrvemfont{#1}}%
```

`\glsfirstlongemfont` Only used by the “long-em” styles.

```
6078 \newcommand*{\glsfirstlongemfont}[1]{\glslongemfont{#1}}%
```

`\glslongemfont` Only used by the “long-em” styles.

```
6079 \newcommand*{\glslongemfont}[1]{\emph{#1}}%
```

`long-short-em`

```
6080 \newabbreviationstyle{long-short-em}%
6081 {%
6082   \GlsXtrUseAbbrStyleSetup{long-short}%
6083 }%
6084 {%
```

Mostly as long-short style:

```
6085   \GlsXtrUseAbbrStyleFmts{long-short}%
6086   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6087 }
```

`long-short-em-desc`

```
6088 \newabbreviationstyle{long-short-em-desc}%
6089 {%
6090   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
6091 }%
6092 {%
```

Mostly as long-short-desc style:

```
6093   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
6094   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6095 }
```

`long-em-short-em`

```
6096 \newabbreviationstyle{long-em-short-em}%
6097 {%
```

`\glslongemfont` is used in the description since `\glsdesc` doesn't set the style.

```
6098   \renewcommand*{\CustomAbbreviationFields}{%
6099     name={\protect\glsabbrvfont{\the\glsshorttok}},
6100     sort={\the\glsshorttok},
6101     first={\protect\glsfirstlongfont{\the\glslongtok}}%
6102     \protect\glsxtrfullsep{\the\glslabeltok}}%
```

```

6103      (\protect\glsfirstabbrvfont{\the\glsshorttok}}),%
6104      firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%
6105      \protect\glstrfullsep{\the\glslabeltok}%
6106      (\protect\glsfirstabbrvfont{\the\glsshortpltok}}),%
6107      plural={\protect\glsabbvfont{\the\glsshortpltok}},%
6108      description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

6109 \renewcommand*\GlsXtrPostNewAbbreviation{%
6110   \glshasattribute{\the\glslabeltok}{regular}%
6111   {%
6112     \glssetattribute{\the\glslabeltok}{regular}{false}%
6113   }%
6114   {}}%
6115 }%
6116 }%
6117 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6118 \GlsXtrUseAbbrStyleFmts{long-short}%
6119 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6120 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6121 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
6122 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
6123 }

```

m-short-em-desc

```

6124 \newabbreviationstyle{long-em-short-em-desc}%
6125 {%
6126   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
6127 }%
6128 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6129 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
6130 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6131 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6132 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
6133 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
6134 }

```

short-em-long Now the short (long) version

```

6135 \newabbreviationstyle{short-em-long}%
6136 {%
6137   \GlsXtrUseAbbrStyleSetup{short-long}%
6138 }%
6139 {%

```

Mostly as short-long style:

```

6140 \GlsXtrUseAbbrStyleFmts{short-long}%
6141 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%

```

```

6142 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6143 }

```

rt-em-long-desc As before but user provides description

```

6144 \newabbreviationstyle{short-em-long-desc}%
6145 {%
6146 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
6147 }%
6148 {%

```

Mostly as short-long-desc style:

```

6149 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
6150 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6151 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6152 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
6153 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
6154 }

```

hort-em-long-em

```

6155 \newabbreviationstyle{short-em-long-em}%
6156 {%

```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```

6157 \renewcommand*{\CustomAbbreviationFields}{%
6158 name={\protect\glsabbrvfont{\the\glsshorttok}},
6159 sort={\the\glsshorttok},
6160 description={\protect\glslongemfont{\the\glslongtok}},%
6161 first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6162 \protect\glsxtrfullsep{\the\glslabeltok}%
6163 (\protect\glsfirstlongfont{\the\glslongtok}}),%
6164 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6165 \protect\glsxtrfullsep{\the\glslabeltok}%
6166 (\protect\glsfirstlongfont{\the\glslongpltok}}),%
6167 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

6168 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6169 \glshasattribute{\the\glslabeltok}{regular}%
6170 {%
6171 \glssetattribute{\the\glslabeltok}{regular}{false}%
6172 }%
6173 {}%
6174 }%
6175 }%
6176 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6177 \GlsXtrUseAbbrStyleFmts{short-long}%
6178 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6179 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6180 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%

```

```

6181 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
6182 }

```

em-long-em-desc

```

6183 \newabbreviationstyle{short-em-long-em-desc}%
6184 {%
6185 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
6186 }%
6187 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6188 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
6189 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6190 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
6191 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
6192 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
6193 }

```

short-em

```

6194 \newabbreviationstyle{short-em}%
6195 {%
6196 \GlsXtrUseAbbrStyleSetup{short-nolong}%
6197 }%
6198 {%

```

Mostly as short style:

```

6199 \GlsXtrUseAbbrStyleFmts{short-nolong}%
6200 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6201 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
6202 }

```

short-em-nolong

```

6203 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

6204 \newabbreviationstyle{short-em-desc}%
6205 {%
6206 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
6207 }%
6208 {%

```

Mostly as short style:

```

6209 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
6210 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
6211 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
6212 }

```

-em-nolong-desc

```

6213 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

long-noshort-em The short form is explicitly invoked through commands like \glssshort.

```
6214 \newabbreviationstyle{long-noshort-em}%  
6215 {%  
6216   \GlsXtrUseAbbrStyleSetup{long-noshort}%  
6217 }%  
6218 {%
```

Mostly as long-noshort style:

```
6219   \GlsXtrUseAbbrStyleFmts{long-noshort}%  
6220   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%  
6221   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%  
6222 }
```

long-em Backward compatibility:

```
6223 \@glsxtr@deprecated@abbrstyle{long-em}{long-noshort-em}
```

g-em-noshort-em The short form is explicitly invoked through commands like \glssshort.

```
6224 \newabbreviationstyle{long-em-noshort-em}%  
6225 {%  
6226   \renewcommand*{\CustomAbbreviationFields}{%  
6227     name={\protect\glsabbrvfont{\the\glssshorttok}},  
6228     sort={\the\glssshorttok},  
6229     first={\protect\glsfirstlongfont{\the\glslongtok}},  
6230     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},  
6231     text={\the\glslongtok},  
6232     plural={\the\glslongpltok},%  
6233     description={\protect\glslongemfont{\the\glslongtok}}%  
6234   }%  
6235   \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
6236     \glssetattribute{\the\glslabeltok}{regular}{true}}%  
6237 }%  
6238 {%
```

Mostly as long-noshort style:

```
6239   \GlsXtrUseAbbrStyleFmts{long-noshort}%  
6240   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%  
6241   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%  
6242   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%  
6243   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%  
6244 }
```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like \glssshort.

```
6245 \newabbreviationstyle{long-noshort-em-desc}%  
6246 {%  
6247   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%  
6248 }%  
6249 {%
```

Mostly as long style:

```
6250 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
6251 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6252 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6253 }
```

long-desc-em Backward compatibility:

```
6254 \@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.

```
6255 \newabbreviationstyle{long-em-noshort-em-desc}%
6256 {%
6257 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
6258 }%
6259 {%
```

Mostly as long style:

```
6260 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
6261 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6262 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6263 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
6264 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
6265 }
```

ort-em-footnote

```
6266 \newabbreviationstyle{short-em-footnote}%
6267 {%
6268 \GlsXtrUseAbbrStyleSetup{short-footnote}%
6269 }%
6270 {%
```

Mostly as long style:

```
6271 \GlsXtrUseAbbrStyleFmts{short-footnote}%
6272 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6273 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6274 }
```

footnote-em Backward compatibility:

```
6275 \@glsxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}
```

em-postfootnote

```
6276 \newabbreviationstyle{short-em-postfootnote}%
6277 {%
6278 \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
6279 }%
6280 {%
```

Mostly as long style:

```
6281 \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
6282 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
6283 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
6284 }
```

postfootnote-em Backward compatibility:

```
6285 \@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.

```
6286 \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.

```
6287 \ifdef\glscurrentfieldvalue
6288 {
6289   \newcommand*\glsxtruserparen[2]{%
6290     \glsxtrfullsep{#2}%
6291     (#1\ifglshasfield{\glsxtruserfield}{#2}{, \glscurrentfieldvalue}{})%
6292   }
6293 }
6294 {
6295   \newcommand*\glsxtruserparen[2]{%
6296     \glsxtrfullsep{#2}%
6297     (#1\ifglshasfield{\glsxtruserfield}{#2}{, \@glo@thisvalue}{})%
6298   }
6299 }
```

Font used for short form:

lsabbrvuserfont

```
6300 \newcommand*\glsabbrvuserfont[1]{#1}
```

Font used for short form on first use:

stabbrvuserfont

```
6301 \newcommand*\glsfirstabbrvuserfont[1]{\glsabbrvuserfont{#1}}
```

Font used for long form:

glslonguserfont

```
6302 \newcommand*\glslonguserfont[1]{#1}
```

Font used for long form on first use:

rstlonguserfont

```
6303 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
6304 \newcommand*{\glsxtrusersuffix}{\glsxtrabbrvpluralsuffix}
```

long-short-user

```
6305 \newabbreviationstyle{long-short-user}%
```

```
6306 {%
```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```
6307 \renewcommand*{\CustomAbbreviationFields}{%
```

```
6308   name={\protect\glsabbrvfont{\the\glsshorttok}},
```

```
6309   sort={\the\glsshorttok},
```

```
6310   first={\protect\glsfirstlongfont{\the\glslongtok}}%
```

```
6311   \protect\glsxtruserparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}{\the\glslabeltok}},
```

```
6312   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%
```

```
6313   \protect\glsxtruserparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}{\the\glslabeltok}},
```

```
6314   plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
```

```
6315   description={\protect\glslonguserfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
6316 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
6317   \glsasattribute{\the\glslabeltok}{regular}%
```

```
6318   {%
```

```
6319     \glssetattribute{\the\glslabeltok}{regular}{false}%
```

```
6320   }%
```

```
6321   {}%
```

```
6322 }%
```

```
6323 }%
```

```
6324 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6325 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
```

```
6326 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
```

```
6327 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
```

```
6328 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
```

```
6329 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
6330 \renewcommand*{\glsxtrfullformat}[2]{%
```

```
6331   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
```

```
6332   \ifglsxtrinsertinside\else##2\fi
```

```
6333   \glsxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
```

```
6334 }%
```

```
6335 \renewcommand*{\glsxtrfullplformat}[2]{%
```

```
6336   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
```

```
6337   \ifglsxtrinsertinside\else##2\fi
```

```
6338   \glsxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
```

```

6339 }%
6340 \renewcommand*{\Glsxtrfullformat}[2]{%
6341   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6342   \ifglxtrinsertinside\else##2\fi
6343   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
6344 }%
6345 \renewcommand*{\Glsxtrfullplformat}[2]{%
6346   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6347   \ifglxtrinsertinside\else##2\fi
6348   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
6349 }%
6350 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

6351 \newabbreviationstyle{long-postshort-user}%
6352 {%
6353   \renewcommand*{\CustomAbbreviationFields}{%
6354     name={\protect\glsabbrvfont{\the\glsshorttok}},
6355     sort={\the\glsshorttok},
6356     first={\protect\glsfirstlongfont{\the\glslongtok}},%
6357     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},%
6358     plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6359     description={\protect\glslonguserfont{\the\glslongtok}}}%
6360 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6361   \csdef{glxtrpostlink\glscategorylabel}{%
6362     \glxtrifwasfirstuse
6363     {%
6364       \glxtruserparen
6365         {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
6366         {\glslabel}%
6367     }%
6368   }%
6369 }%
6370 \glshasattribute{\the\glslabeltok}{regular}%
6371 {%
6372   \glissetattribute{\the\glslabeltok}{regular}{false}%
6373 }%
6374 }%
6375 }%
6376 }%
6377 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6378 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
6379 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
6380 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
6381 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
6382 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

6383 \renewcommand*{\glxtrfullformat}[2]{%
6384   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6385   \ifglxtrinsertinside\else##2\fi
6386 }%
6387 \renewcommand*{\glxtrfullplformat}[2]{%
6388   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6389   \ifglxtrinsertinside\else##2\fi
6390 }%
6391 \renewcommand*{\Glsxtrfullformat}[2]{%
6392   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6393   \ifglxtrinsertinside\else##2\fi
6394 }%
6395 \renewcommand*{\Glsxtrfullplformat}[2]{%
6396   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6397   \ifglxtrinsertinside\else##2\fi
6398 }%

```

In-line format:

```

6399 \renewcommand*{\glxtrinlinefullformat}[2]{%
6400   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6401   \ifglxtrinsertinside\else##2\fi
6402   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
6403 }%
6404 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6405   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6406   \ifglxtrinsertinside\else##2\fi
6407   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
6408 }%
6409 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6410   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6411   \ifglxtrinsertinside\else##2\fi
6412   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
6413 }%
6414 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6415   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6416   \ifglxtrinsertinside\else##2\fi
6417   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
6418 }%
6419 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

6420 \newabbreviationstyle{long-postshort-user-desc}%
6421 {%
6422   \renewcommand*{\CustomAbbreviationFields}{%
6423     name={\protect\glsfirstlongfont{\the\glslongtok}%
6424           \protect\glxtruserparen
6425           {\protect\glsabbrvfont{\the\glsshorttok}}{\the\glslabeltok}},
6426     sort={\the\glslongtok},
6427     first={\protect\glsfirstlongfont{\the\glslongtok}},%
6428     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},%

```

```

6429 plural={\protect\glsabrvfont{\the\glsshortpltok}}}%
6430 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6431   \csdef{glsxtrpostlink\glscategorylabel}{%
6432     \glsxtrifwasfirstuse
6433     {%
6434       \glsxtruserparen
6435       {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
6436       {\glslabel}%
6437     }%
6438   }%
6439 }%
6440 \glsattribute{\the\glslabeltok}{regular}%
6441 {%
6442   \glssetattribute{\the\glslabeltok}{regular}{false}%
6443 }%
6444 {}%
6445 }%
6446 }%
6447 {%
6448   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
6449 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

6450 \newabbreviationstyle{short-postlong-user}%
6451 {%
6452   \renewcommand*{\CustomAbbreviationFields}{%
6453     name={\protect\glsabrvfont{\the\glsshorttok}},
6454     sort={\the\glsshorttok},
6455     first={\protect\glsfirstlongfont{\the\glslongtok}},%
6456     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},%
6457     plural={\protect\glsabrvfont{\the\glsshortpltok}},%
6458     description={\protect\glslonguserfont{\the\glslongtok}}}%
6459 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6460   \csdef{glsxtrpostlink\glscategorylabel}{%
6461     \glsxtrifwasfirstuse
6462     {%
6463       \glsxtruserparen
6464       {\glsfirstabbrvuserfont{\glsentrylong{\glslabel}}}%
6465       {\glslabel}%
6466     }%
6467   }%
6468 }%
6469 \glsattribute{\the\glslabeltok}{regular}%
6470 {%
6471   \glssetattribute{\the\glslabeltok}{regular}{false}%
6472 }%
6473 {}%
6474 }%
6475 }%

```

6476 {%

In case the user wants to mix and match font styles, these are redefined here.

```
6477 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
6478 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
6479 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
6480 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
6481 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

First use full form:

```
6482 \renewcommand*{\glxtrfullformat}[2]{%
6483   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6484   \ifglxtrininsertinside\else##2\fi
6485 }%
6486 \renewcommand*{\glxtrfullplformat}[2]{%
6487   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6488   \ifglxtrininsertinside\else##2\fi
6489 }%
6490 \renewcommand*{\Glsxtrfullformat}[2]{%
6491   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6492   \ifglxtrininsertinside\else##2\fi
6493 }%
6494 \renewcommand*{\Glsxtrfullplformat}[2]{%
6495   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6496   \ifglxtrininsertinside\else##2\fi
6497 }%
```

In-line format:

```
6498 \renewcommand*{\glxtrinlinefullformat}[2]{%
6499   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6500   \ifglxtrininsertinside\else##2\fi
6501   \glxtruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%
6502 }%
6503 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6504   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6505   \ifglxtrininsertinside\else##2\fi
6506   \glxtruserparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}{##1}%
6507 }%
6508 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6509   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6510   \ifglxtrininsertinside\else##2\fi
6511   \glxtruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%
6512 }%
6513 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6514   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6515   \ifglxtrininsertinside\else##2\fi
6516   \glxtruserparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}{##1}%
6517 }%
6518 }
```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```
6519 \newabbreviationstyle{short-postlong-user-desc}%
6520 {%
6521   \renewcommand*{\CustomAbbreviationFields}{%
6522     name={\protect\glsabbrvfont{\the\glsshorttok}%
6523       \protect\glsxtruserparen
6524         {\protect\glsfirstlongfont{\the\glslongpltok}}%
6525         {\the\glslabeltok}},
6526     sort={\the\glsshorttok},
6527     first={\protect\glsfirstlongfont{\the\glslongtok}},%
6528     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},%
6529     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6530 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6531   \csdef{glsxtrpostlink\glscategorylabel}{%
6532     \glsxtrifwasfirstuse
6533     {%
6534       \glsxtruserparen
6535       {\glsfirstabbrvuserfont{\glsentrylong{\glslabel}}}%
6536       {\glslabel}%
6537     }%
6538   }%
6539 }%
6540 \glsattribute{\the\glslabeltok}{regular}%
6541 {%
6542   \glssetattribute{\the\glslabeltok}{regular}{false}%
6543 }%
6544 {}%
6545 }%
6546 }%
6547 {%
6548   \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
6549 }
```

short-user-desc

```
6550 \newabbreviationstyle{long-short-user-desc}%
6551 {%
6552   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
6553 }%
6554 {%
6555   \GlsXtrUseAbbrStyleFmts{long-short-user}%
6556 }
```

short-long-user

```
6557 \newabbreviationstyle{short-long-user}%
6558 {%
6559   \glslonguserfont is used in the description since \glsdesc doesn't set the style.
6559   \renewcommand*{\CustomAbbreviationFields}{%
6560     name={\protect\glsabbrvfont{\the\glsshorttok}},
```

```

6561     sort={\the\glsshorttok},
6562     description={\protect\glslonguserfont{\the\glslongtok}},%
6563     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6564     \protect\glxtruserparen{\protect\glslonguserfont{\the\glslongtok}}{\the\glslabeltok}},%
6565     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6566     \protect\glxtruserparen{\protect\glslonguserfont{\the\glslongpltok}}{\the\glslabeltok}},%
6567     plural={\protect\glsabbvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

6568 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6569   \glsattribute{\the\glslabeltok}{regular}}%
6570   {%
6571     \glsattribute{\the\glslabeltok}{regular}{false}}%
6572   }%
6573   {}%
6574 }%
6575 }%
6576 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6577 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
6578 \renewcommand*{\glsabbvfont[1]}{\glsabbvuserfont{##1}}%
6579 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
6580 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
6581 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6582 \renewcommand*{\glxtrfullformat}[2]{%
6583   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6584   \ifglxtrininsertinside\else##2\fi
6585   \glxtruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%
6586 }%
6587 \renewcommand*{\glxtrfullplformat}[2]{%
6588   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6589   \ifglxtrininsertinside\else##2\fi
6590   \glxtruserparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}{##1}%
6591 }%
6592 \renewcommand*{\Glsxtrfullformat}[2]{%
6593   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6594   \ifglxtrininsertinside\else##2\fi
6595   \glxtruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%
6596 }%
6597 \renewcommand*{\Glsxtrfullplformat}[2]{%
6598   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6599   \ifglxtrininsertinside\else##2\fi
6600   \glxtruserparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}{##1}%
6601 }%
6602 }

```

-long-user-desc

```

6603 \newabbreviationstyle{short-long-user-desc}%
6604 {%
6605   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
6606 }%
6607 {%
6608   \GlsXtrUseAbbrStyleFmts{short-long-user}%
6609 }

```

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 `\glsentryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glsabbrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref's \texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glsxtrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to false, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that glossaries automatically loads `textcase`, so the label can be protected from case change with `textcase's \NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
6610 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```

6611 \renewcommand*{\markright}[1]{%
6612   \glsxtrmarkhook
6613   \@glsxtr@org@markright{\@glsxtrinmark#1\@glsxtrnotinmark}%
6614   \glsxtrrestoremarkhook
6615 }

```

`\markboth` Save original definition:

```
6616 \let\@glsxtr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):


```

6617 \renewcommand*{\markboth}[2]{%
6618   \glxtrmarkhook
6619   \@glxtr@org@markboth
6620   {\@glxtrinmark#1\@glxtrnotinmark}%
6621   {\@glxtrinmark#2\@glxtrnotinmark}%
6622   \glxtrrestoremarkhook
6623 }

```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```

6624 \newcommand*{\glxtrRevertMarks}{%
6625   \let\markright\@glxtr@org@markright
6626   \let\markboth\@glxtr@org@markboth
6627 }

```

\glxtrifinmark

```

6628 \newcommand*{\glxtrifinmark}[2]{#2}

```

\@glxtrinmark

```

6629 \newrobustcmd*{\@glxtrinmark}{%
6630   \let\glxtrifinmark\@firstoftwo
6631 }

```

glxtrnotinmark

```

6632 \newrobustcmd*{\@glxtrnotinmark}{%
6633   \let\glxtrifinmark\@secondoftwo
6634 }

```

\glxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```

6635 \newcommand*{\glxtrmarkhook}{%

```

Save current definitions:

```

6636   \let\@glxtr@org@MakeUppercase\MakeUppercase
6637   \let\@glxtr@org@glxtrtitleshort\glxtrtitleshort
6638   \let\@glxtr@org@glxtrtitleshortpl\glxtrtitleshortpl
6639   \let\@glxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
6640   \let\@glxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
6641   \let\@glxtr@org@glxtrtitletext\glxtrtitletext
6642   \let\@glxtr@org@Glsxtrtitletext\Glsxtrtitletext
6643   \let\@glxtr@org@glxtrtitleplural\glxtrtitleplural
6644   \let\@glxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
6645   \let\@glxtr@org@glxtrtitlefirst\glxtrtitlefirst
6646   \let\@glxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
6647   \let\@glxtr@org@glxtrtitlefirstplural\glxtrtitlefirstplural
6648   \let\@glxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
6649   \let\@glxtr@org@glxtrtitlelong\glxtrtitlelong
6650   \let\@glxtr@org@glxtrtitlelongpl\glxtrtitlelongpl

```

```

6651 \let\@glsxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
6652 \let\@glsxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
6653 \let\@glsxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
6654 \let\@glsxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl
6655 \let\@glsxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
6656 \let\@glsxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl

```

New definitions

```

6657 \let\glsxtrifinmark\@firstoftwo
6658 \let\MakeUppercase\MakeTextUppercase
6659 \let\glsxtrtitleshort\glsxtrheadshort
6660 \let\glsxtrtitleshortpl\glsxtrheadshortpl
6661 \let\Glsxtrtitleshort\Glsxtrheadshort
6662 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
6663 \let\glsxtrtitletext\glsxtrheadtext
6664 \let\Glsxtrtitletext\Glsxtrheadtext
6665 \let\glsxtrtitleplural\glsxtrheadplural
6666 \let\Glsxtrtitleplural\Glsxtrheadplural
6667 \let\glsxtrtitlefirst\glsxtrheadfirst
6668 \let\Glsxtrtitlefirst\Glsxtrheadfirst
6669 \let\glsxtrtitlefirstplural\glsxtrheadfirstplural
6670 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
6671 \let\glsxtrtitlelong\glsxtrheadlong
6672 \let\glsxtrtitlelongpl\glsxtrheadlongpl
6673 \let\Glsxtrtitlelong\Glsxtrheadlong
6674 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
6675 \let\glsxtrtitlefull\glsxtrheadfull
6676 \let\glsxtrtitlefullpl\glsxtrheadfullpl
6677 \let\Glsxtrtitlefull\Glsxtrheadfull
6678 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
6679 }

```

`\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.)

```

6680 \newcommand*{\glsxtrrestoremarkhook}{%
6681 \let\glsxtrifinmark\@secondoftwo
6682 \let\MakeUppercase\@glsxtr@org@MakeUppercase
6683 \let\glsxtrtitleshort\@glsxtr@org@glsxtrtitleshort
6684 \let\glsxtrtitleshortpl\@glsxtr@org@glsxtrtitleshortpl
6685 \let\Glsxtrtitleshort\@glsxtr@org@Glsxtrtitleshort
6686 \let\Glsxtrtitleshortpl\@glsxtr@org@Glsxtrtitleshortpl
6687 \let\glsxtrtitletext\@glsxtr@org@glsxtrtitletext
6688 \let\Glsxtrtitletext\@glsxtr@org@Glsxtrtitletext
6689 \let\glsxtrtitleplural\@glsxtr@org@glsxtrtitleplural
6690 \let\Glsxtrtitleplural\@glsxtr@org@Glsxtrtitleplural
6691 \let\glsxtrtitlefirst\@glsxtr@org@glsxtrtitlefirst
6692 \let\Glsxtrtitlefirst\@glsxtr@org@Glsxtrtitlefirst

```

```

6693 \let\glsxtrtitlefirstplural\@glsxtr@org@glsxtrtitlefirstplural
6694 \let\Glsxtrtitlefirstplural\@glsxtr@org@Glsxtrtitlefirstplural
6695 \let\glsxtrtitlelong\@glsxtr@org@glsxtrtitlelong
6696 \let\glsxtrtitlelongpl\@glsxtr@org@glsxtrtitlelongpl
6697 \let\Glsxtrtitlelong\@glsxtr@org@Glsxtrtitlelong
6698 \let\Glsxtrtitlelongpl\@glsxtr@org@Glsxtrtitlelongpl
6699 \let\glsxtrtitlefull\@glsxtr@org@glsxtrtitlefull
6700 \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
6701 \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
6702 \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl
6703 }

```

Instead of using one document-wide conditional, use headuc attribute to determine whether or not to use the all upper case form.

glsxtrheadshort Command used to display short form in the page header.

```

6704 \newcommand*{\glsxtrheadshort}[1]{%
6705   \protect\NoCaseChange
6706   {%
6707     \glsifattribute{#1}{headuc}{true}%
6708     {%
6709       \GLSxtrshort[noindex,hyper=false]{#1}[]%
6710     }%
6711     {%
6712       \glsxtrshort[noindex,hyper=false]{#1}[]%
6713     }%
6714   }%
6715 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

6716 \newrobustcmd*{\lsxtrtitleshort}[1]{%
6717   \glsxtrshort[noindex,hyper=false]{#1}[]%
6718 }

```

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.

```

6719 \newcommand*{\glsxtrheadshortpl}[1]{%
6720   \protect\NoCaseChange
6721   {%
6722     \glsifattribute{#1}{headuc}{true}%
6723     {%
6724       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
6725     }%
6726     {%
6727       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
6728     }%
6729   }%
6730 }

```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents.

```
6731 \newrobustcmd*{\glxtrtitleshortpl}[1]{%
6732   \glxtrshortpl[noindex,hyper=false]{#1}[]%
6733 }
```

`Glsxtrheadshort` Command used to display short form in the page header with the first letter converted to upper case.

```
6734 \newcommand*{\Glsxtrheadshort}[1]{%
6735   \protect\NoCaseChange
6736   {%
6737     \glsifattribute{#1}{headuc}{true}%
6738     {%
6739       \GLSxtrshort[noindex,hyper=false]{#1}[]%
6740     }%
6741     {%
6742       \Glsxtrshort[noindex,hyper=false]{#1}[]%
6743     }%
6744   }%
6745 }
```

`lsxtrtitleshort` Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
6746 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
6747   \Glsxtrshort[noindex,hyper=false]{#1}[]%
6748 }
```

`sxtrheadshortpl` Command used to display plural short form in the page header with the first letter converted to upper case.

```
6749 \newcommand*{\Glsxtrheadshortpl}[1]{%
6750   \protect\NoCaseChange
6751   {%
6752     \glsifattribute{#1}{headuc}{true}%
6753     {%
6754       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
6755     }%
6756     {%
6757       \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
6758     }%
6759   }%
6760 }
```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
6761 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
6762   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
6763 }
```

`\glxtrheadtext` As above but for the text value.

```

6764 \newcommand*{\glxtrheadtext}[1]{%
6765   \protect\NoCaseChange
6766   {%
6767     \gl@ifattribute{#1}{headuc}{true}%
6768     {%
6769       \GLStext[noindex,hyper=false]{#1}[]%
6770     }%
6771     {%
6772       \glstext[noindex,hyper=false]{#1}[]%
6773     }%
6774   }%
6775 }

```

glxtrtitletext Command to display text value in section title and table of contents.

```

6776 \newrobustcmd*{\glxtrtitletext}[1]{%
6777   \glstext[noindex,hyper=false]{#1}[]%
6778 }

```

\Glsxtrheadtext First letter converted to upper case

```

6779 \newcommand*{\Glsxtrheadtext}[1]{%
6780   \protect\NoCaseChange
6781   {%
6782     \gl@ifattribute{#1}{headuc}{true}%
6783     {%
6784       \GLStext[noindex,hyper=false]{#1}[]%
6785     }%
6786     {%
6787       \Glstext[noindex,hyper=false]{#1}[]%
6788     }%
6789   }%
6790 }

```

Glsxtrtitletext Command to display text value in section title and table of contents with the first letter changed to upper case.

```

6791 \newrobustcmd*{\Glsxtrtitletext}[1]{%
6792   \Glstext[noindex,hyper=false]{#1}[]%
6793 }

```

lsxtrheadplural As above but for the plural value.

```

6794 \newcommand*{\lsxtrheadplural}[1]{%
6795   \protect\NoCaseChange
6796   {%
6797     \gl@ifattribute{#1}{headuc}{true}%
6798     {%
6799       \GLSplural[noindex,hyper=false]{#1}[]%
6800     }%
6801     {%
6802       \glsplural[noindex,hyper=false]{#1}[]%
6803     }%

```

```

6804 }%
6805 }

sxttrtitleplural Command to display plural value in section title and table of contents.
6806 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
6807   \Glsplural[noindex,hyper=false]{#1}[]%
6808 }

lsxtrheadplural Convert first letter to upper case.
6809 \newcommand*{\Glsxtrheadplural}[1]{%
6810   \protect\NoCaseChange
6811   {%
6812     \glsifattribute{#1}{headuc}{true}%
6813     {%
6814       \GLSplural[noindex,hyper=false]{#1}[]%
6815     }%
6816     {%
6817       \Glsplural[noindex,hyper=false]{#1}[]%
6818     }%
6819   }%
6820 }

sxttrtitleplural Command to display plural value in section title and table of contents with the first letter
changed to upper case.
6821 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
6822   \Glsplural[noindex,hyper=false]{#1}[]%
6823 }

glxtrheadfirst As above but for the first value.
6824 \newcommand*{\glxtrheadfirst}[1]{%
6825   \protect\NoCaseChange
6826   {%
6827     \glsifattribute{#1}{headuc}{true}%
6828     {%
6829       \GLSfirst[noindex,hyper=false]{#1}[]%
6830     }%
6831     {%
6832       \glsfirst[noindex,hyper=false]{#1}[]%
6833     }%
6834   }%
6835 }

lsxtrtitlefirst Command to display first value in section title and table of contents.
6836 \newrobustcmd*{\glxtrtitlefirst}[1]{%
6837   \glsfirst[noindex,hyper=false]{#1}[]%
6838 }

Glsxtrheadfirst First letter converted to upper case

```

```

6839 \newcommand*{\Glsxtrheadfirst}[1]{%
6840   \protect\NoCaseChange
6841   {%
6842     \glsifattribute{#1}{headuc}{true}%
6843     {%
6844       \GLSfirst[noindex,hyper=false]{#1}[]%
6845     }%
6846     {%
6847       \Glsfirst[noindex,hyper=false]{#1}[]%
6848     }%
6849   }%
6850 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```

6851 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
6852   \GLSfirst[noindex,hyper=false]{#1}[]%
6853 }

```

headfirstplural As above but for the firstplural value.

```

6854 \newcommand*{\Glsxtrheadfirstplural}[1]{%
6855   \protect\NoCaseChange
6856   {%
6857     \glsifattribute{#1}{headuc}{true}%
6858     {%
6859       \GLSfirstplural[noindex,hyper=false]{#1}[]%
6860     }%
6861     {%
6862       \glsfirstplural[noindex,hyper=false]{#1}[]%
6863     }%
6864   }%
6865 }

```

itlefirstplural Command to display firstplural value in section title and table of contents.

```

6866 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
6867   \glsfirstplural[noindex,hyper=false]{#1}[]%
6868 }

```

headfirstplural First letter converted to upper case

```

6869 \newcommand*{\Glsxtrheadfirstplural}[1]{%
6870   \protect\NoCaseChange
6871   {%
6872     \glsifattribute{#1}{headuc}{true}%
6873     {%
6874       \GLSfirstplural[noindex,hyper=false]{#1}[]%
6875     }%
6876     {%
6877       \Glsfirstplural[noindex,hyper=false]{#1}[]%
6878     }%

```

```
6879 }%
6880 }
```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
6881 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
6882   \Glsfirstplural[noindex,hyper=false]{#1}[]%
6883 }
```

`\glxstrheadlong` Command used to display long form in the page header.

```
6884 \newcommand*{\glxstrheadlong}[1]{%
6885   \protect\NoCaseChange
6886   {%
6887     \glsifattribute{#1}{headuc}{true}%
6888     {%
6889       \GLSxtrlong[noindex,hyper=false]{#1}[]%
6890     }%
6891     {%
6892       \glxstrlong[noindex,hyper=false]{#1}[]%
6893     }%
6894   }%
6895 }
```

`\glxstrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
6896 \newrobustcmd*{\glxstrtitlelong}[1]{%
6897   \glxstrlong[noindex,hyper=false]{#1}[]%
6898 }
```

`\glxstrheadlongpl` 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.

```
6899 \newcommand*{\glxstrheadlongpl}[1]{%
6900   \protect\NoCaseChange
6901   {%
6902     \glsifattribute{#1}{headuc}{true}%
6903     {%
6904       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
6905     }%
6906     {%
6907       \glxstrlongpl[noindex,hyper=false]{#1}[]%
6908     }%
6909   }%
6910 }
```

`\glxstrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```
6911 \newrobustcmd*{\glxstrtitlelongpl}[1]{%
6912   \glxstrlongpl[noindex,hyper=false]{#1}[]%
6913 }
```


`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```
6914 \newcommand*{\Glsxtrheadlong}[1]{%
6915   \protect\NoCaseChange
6916   {%
6917     \glsifattribute{#1}{headuc}{true}%
6918     {%
6919       \Glsxtrlong[noindex,hyper=false]{#1}[]%
6920     }%
6921     {%
6922       \Glsxtrlong[noindex,hyper=false]{#1}[]%
6923     }%
6924   }%
6925 }
```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
6926 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
6927   \Glsxtrlong[noindex,hyper=false]{#1}[]%
6928 }
```

`lGlsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```
6929 \newcommand*{\Glsxtrheadlongpl}[1]{%
6930   \protect\NoCaseChange
6931   {%
6932     \glsifattribute{#1}{headuc}{true}%
6933     {%
6934       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
6935     }%
6936     {%
6937       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
6938     }%
6939   }%
6940 }
```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
6941 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
6942   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
6943 }
```

`\glsxtrheadfull` Command used to display full form in the page header.

```
6944 \newcommand*{\glsxtrheadfull}[1]{%
6945   \protect\NoCaseChange
6946   {%
6947     \glsifattribute{#1}{headuc}{true}%
6948     {%
```

```

6949 \GLSxtrfull[noindex,hyper=false]{#1}[]%
6950 }%
6951 {%
6952 \glsxtrfull[noindex,hyper=false]{#1}[]%
6953 }%
6954 }%
6955 }

```

glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents.

```

6956 \newrobustcmd*{\glsxtrtitlefull}[1]{%
6957 \glsxtrfull[noindex,hyper=false]{#1}[]%
6958 }

```

glsxtrheadfullpl Command used to display plural full form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrfullpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

6959 \newcommand*{\glsxtrheadfullpl}[1]{%
6960 \protect\NoCaseChange
6961 {%
6962 \glsifattribute{#1}{headuc}{true}%
6963 {%
6964 \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
6965 }%
6966 {%
6967 \glsxtrfullpl[noindex,hyper=false]{#1}[]%
6968 }%
6969 }%
6970 }

```

glsxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents.

```

6971 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
6972 \glsxtrfullpl[noindex,hyper=false]{#1}[]%
6973 }

```

\Glsxtrheadfull Command used to display full form in the page header with the first letter converted to upper case.

```

6974 \newcommand*{\Glsxtrheadfull}[1]{%
6975 \protect\NoCaseChange
6976 {%
6977 \glsifattribute{#1}{headuc}{true}%
6978 {%
6979 \GLSxtrfull[noindex,hyper=false]{#1}[]%
6980 }%
6981 {%
6982 \Glsxtrfull[noindex,hyper=false]{#1}[]%
6983 }%
6984 }%
6985 }

```

`\Glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
6986 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
6987   \Glsxtrfull[noindex,hyper=false]{#1}[]%
6988 }
```

`\Glsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```
6989 \newcommand*{\Glsxtrheadfullpl}[1]{%
6990   \protect\NoCaseChange
6991   {%
6992     \glsifattribute{#1}{headuc}{true}%
6993     {%
6994       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
6995     }%
6996     {%
6997       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
6998     }%
6999   }%
7000 }
```

`\Glsxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
7001 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
7002   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
7003 }
```

`\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.

```
7004 \ifdef\texorpdfstring
7005 {
7006   \newcommand*{\glsfmtshort}[1]{%
7007     \texorpdfstring
7008       {\glsxtrtitleshort{#1}}%
7009       {\glsentryshort{#1}}%
7010   }
7011 }
7012 {
7013   \newcommand*{\glsfmtshort}[1]{%
7014     \glsxtrtitleshort{#1}}
7015 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
7016 \ifdef\texorpdfstring
7017 {
7018   \newcommand*{\glsfmtshortpl}[1]{%
```

```

7019 \texorpdfstring
7020   {\glxtrtitleshortpl{#1}}}%
7021   {\glsentryshortpl{#1}}}%
7022 }
7023 }
7024 {
7025 \newcommand*{\glsfmtshortpl}[1]{%
7026   \glxtrtitleshortpl{#1}}
7027 }

```

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).

```

7028 \ifdef\texorpdfstring
7029 {
7030   \newcommand*{\Glsfmtshort}[1]{%
7031     \texorpdfstring
7032       {\Glsxtrtitleshort{#1}}}%
7033     {\glsentryshort{#1}}}%
7034   }
7035 }
7036 {
7037   \newcommand*{\Glsfmtshort}[1]{%
7038     \Glsxtrtitleshort{#1}}
7039 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

7040 \ifdef\texorpdfstring
7041 {
7042   \newcommand*{\Glsfmtshortpl}[1]{%
7043     \texorpdfstring
7044       {\Glsxtrtitleshortpl{#1}}}%
7045     {\glsentryshortpl{#1}}}%
7046   }
7047 }
7048 {
7049   \newcommand*{\Glsfmtshortpl}[1]{%
7050     \Glsxtrtitleshortpl{#1}}
7051 }

```

`\glsfmttext` As above but for the text value.

```

7052 \ifdef\texorpdfstring
7053 {
7054   \newcommand*{\glsfmttext}[1]{%
7055     \texorpdfstring
7056       {\glxtrtitletext{#1}}}%
7057     {\glsentrytext{#1}}}%
7058   }

```

```

7059 }
7060 {
7061   \newcommand*{\glsfmttext}[1]{%
7062     \glsxtrtitletext{#1}}
7063 }

```

`\Glsfmttext` First letter converted to upper case.

```

7064 \ifdef\texorpdfstring
7065 {
7066   \newcommand*{\Glsfmttext}[1]{%
7067     \texorpdfstring
7068     {\Glsxtrtitletext{#1}}%
7069     {\glsentrytext{#1}}%
7070   }
7071 }
7072 {
7073   \newcommand*{\Glsfmttext}[1]{%
7074     \Glsxtrtitletext{#1}}
7075 }

```

`\glsfmtplural` As above but for the plural value.

```

7076 \ifdef\texorpdfstring
7077 {
7078   \newcommand*{\glsfmtplural}[1]{%
7079     \texorpdfstring
7080     {\glsxtrtitleplural{#1}}%
7081     {\glsentryplural{#1}}%
7082   }
7083 }
7084 {
7085   \newcommand*{\glsfmtplural}[1]{%
7086     \glsxtrtitleplural{#1}}
7087 }

```

`\Glsfmtplural` First letter converted to upper case.

```

7088 \ifdef\texorpdfstring
7089 {
7090   \newcommand*{\Glsfmtplural}[1]{%
7091     \texorpdfstring
7092     {\Glsxtrtitleplural{#1}}%
7093     {\glsentryplural{#1}}%
7094   }
7095 }
7096 {
7097   \newcommand*{\Glsfmtplural}[1]{%
7098     \Glsxtrtitleplural{#1}}
7099 }

```

`\glsfmtfirst` As above but for the first value.

```

7100 \ifdef\teorpdfstring
7101 {
7102   \newcommand*\glsfmtfirst}[1]{%
7103     \teorpdfstring
7104     {\glstrtitlefirst{#1}}%
7105     {\glentryfirst{#1}}%
7106   }
7107 }
7108 {
7109   \newcommand*\glsfmtfirst}[1]{%
7110     \glstrtitlefirst{#1}}
7111 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

7112 \ifdef\teorpdfstring
7113 {
7114   \newcommand*\Glsfmtfirst}[1]{%
7115     \teorpdfstring
7116     {\Glsstrtitlefirst{#1}}%
7117     {\glentryfirst{#1}}%
7118   }
7119 }
7120 {
7121   \newcommand*\Glsfmtfirst}[1]{%
7122     \Glsstrtitlefirst{#1}}
7123 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

7124 \ifdef\teorpdfstring
7125 {
7126   \newcommand*\glsfmtfirstpl}[1]{%
7127     \teorpdfstring
7128     {\glstrtitlefirstplural{#1}}%
7129     {\glentryfirstplural{#1}}%
7130   }
7131 }
7132 {
7133   \newcommand*\glsfmtfirstpl}[1]{%
7134     \glstrtitlefirstplural{#1}}
7135 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

7136 \ifdef\teorpdfstring
7137 {
7138   \newcommand*\Glsfmtfirstpl}[1]{%
7139     \teorpdfstring
7140     {\Glsstrtitlefirstplural{#1}}%
7141     {\glentryfirstplural{#1}}%
7142   }

```

```

7143 }
7144 {
7145   \newcommand*{\Glsfmtfirstpl}[1]{%
7146     \Glsxtrtitlefirstplural{#1}}
7147 }

```

`\glsfmtlong` As above but for the long value.

```

7148 \ifdef\texorpdfstring
7149 {
7150   \newcommand*{\glsfmtlong}[1]{%
7151     \texorpdfstring
7152     {\glsxtrtitlelong{#1}}%
7153     {\glsentrylong{#1}}%
7154   }
7155 }
7156 {
7157   \newcommand*{\glsfmtlong}[1]{%
7158     \glsxtrtitlelong{#1}}
7159 }

```

`\Glsfmtlong` First letter converted to upper case.

```

7160 \ifdef\texorpdfstring
7161 {
7162   \newcommand*{\Glsfmtlong}[1]{%
7163     \texorpdfstring
7164     {\Glsxtrtitlelong{#1}}%
7165     {\glsentrylong{#1}}%
7166   }
7167 }
7168 {
7169   \newcommand*{\Glsfmtlong}[1]{%
7170     \Glsxtrtitlelong{#1}}
7171 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

7172 \ifdef\texorpdfstring
7173 {
7174   \newcommand*{\glsfmtlongpl}[1]{%
7175     \texorpdfstring
7176     {\glsxtrtitlelongpl{#1}}%
7177     {\glsentrylongpl{#1}}%
7178   }
7179 }
7180 {
7181   \newcommand*{\glsfmtlongpl}[1]{%
7182     \glsxtrtitlelongpl{#1}}
7183 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

7184 \ifdef\txorpdfstring
7185 {
7186   \newcommand*\Glsfmtlongpl}[1]{%
7187     \txorpdfstring
7188     {\Glsxtrtitlelongpl{#1}}%
7189     {\glsentrylongpl{#1}}%
7190   }
7191 }
7192 {
7193   \newcommand*\Glsfmtlongpl}[1]{%
7194     \Glsxtrtitlelongpl{#1}}
7195 }

```

`\glsfmtfull` In-line full format.

```

7196 \ifdef\txorpdfstring
7197 {
7198   \newcommand*\glsfmtfull}[1]{%
7199     \txorpdfstring
7200     {\glsxtrtitlefull{#1}}%
7201     {\glsxtrinlinefullformat{#1}{}}%
7202   }
7203 }
7204 {
7205   \newcommand*\glsfmtfull}[1]{%
7206     \glsxtrtitlefull{#1}}
7207 }

```

`\Glsfmtfull` First letter converted to upper case.

```

7208 \ifdef\txorpdfstring
7209 {
7210   \newcommand*\Glsfmtfull}[1]{%
7211     \txorpdfstring
7212     {\Glsxtrtitlefull{#1}}%
7213     {\Glsxtrinlinefullformat{#1}{}}%
7214   }
7215 }
7216 {
7217   \newcommand*\Glsfmtfull}[1]{%
7218     \Glsxtrtitlefull{#1}}
7219 }

```

`\glsfmtfullpl` In-line full plural format.

```

7220 \ifdef\txorpdfstring
7221 {
7222   \newcommand*\glsfmtfullpl}[1]{%
7223     \txorpdfstring
7224     {\glsxtrtitlefullpl{#1}}%
7225     {\glsxtrinlinefullplformat{#1}{}}%
7226   }

```



```

7227 }
7228 {
7229   \newcommand*{\Glsfmtfullpl}[1]{%
7230     \Glsxtrtitlefullpl{#1}}
7231 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

7232 \ifdef\texorpdfstring
7233 {
7234   \newcommand*{\Glsfmtfullpl}[1]{%
7235     \texorpdfstring
7236       {\Glsxtrtitlefullpl{#1}}%
7237       {\Glsxtrinlinefullplformat{#1}{}}%
7238   }
7239 }
7240 {
7241   \newcommand*{\Glsfmtfullpl}[1]{%
7242     \Glsxtrtitlefullpl{#1}}
7243 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

7244 \newcommand*{\RequireGlossariesExtraLang}[1]{%
7245   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
7246 }

```

`sariesExtraLang`

```

7247 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
7248   \ProvidesFile{glossariesxtr-#1.ldf}%
7249 }

```

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.)

```

7250 \@ifpackageloaded{tracklang}
7251 {%
7252   \AnyTrackedLanguages
7253   {%
7254     \ForEachTrackedDialect{\this@dialect}{%
7255       \IfTrackedLanguageFileExists{\this@dialect}%
7256       {glossariesxtr-}% prefix
7257       {.ldf}%
7258     }%

```

```

7259         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
7260     }%
7261     {%
7262     }%
7263     }%
7264 }%
7265 {}%
7266 }
7267 {}

```

Load glossaries-extra-stylemods if required.

```
7268 \@glsxtr@redefstyles
```

and set the style:

```
7269 \@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:

```
7270 \NeedsTeXFormat{LaTeX2e}
7271 \ProvidesPackage{glossaries-extra-stylemods}[2017/02/03 v1.12 (NLCT)]
```

Provide package options to automatically load required predefined styles. The simplest method is to just test for the existence of the file `glossary-⟨option⟩.sty`. Packages can't be loaded whilst the options are being processed, so save the list in `\@glxtr@loadstyles`.

`sxtr@loadstyles`

```
7272 \newcommand*{\@glxtr@loadstyles}{%
7273 \DeclareOption*{%
7274   \IfFileExists{glossary-\CurrentOption.sty}%
7275   {\eappto\@glxtr@loadstyles{%
7276     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
7277   {\PackageError{glossaries-extra-styles}%
7278     {Unknown option '\CurrentOption'}{}}
7279 }
```

Process the package options:

```
7280 \ProcessOptions
```

Load the required packages:

```
7281 \@glxtr@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`

```
7282 \providecommand{\renewglossarystyle}[2]{%
7283   \ifcsundef{@glstyle@#1}%
7284   {%
7285     \PackageError{glossaries}{Glossary style '#1' isn't already defined}{}%

```

```

7286 }%
7287 {%
7288   \csdef{@glsstyle@#1}{#2}%
7289 }%
7290 }

```

2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying.

```

7291 \ifdefined{@glsstyle@listdotted}
7292 {%
7293   \renewglossarystyle{listdotted}{%
7294     \setglossarystyle{list}%
7295     \renewcommand*{\glossentry}[2]{%
7296       \item[]\makebox[\glslistdottedwidth][l]{%
7297         \glsentryitem{##1}%
7298         \glstarget{##1}{\glossentryname{##1}}%
7299         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
7300       \glossentrydesc{##1}\glspostdescription}%
7301   \renewcommand*{\subglossentry}[3]{%
7302     \item[]\makebox[\glslistdottedwidth][l]{%
7303       \glssubentryitem{##2}%
7304       \glstarget{##2}{\glossentryname{##2}}%
7305       \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
7306     \glossentrydesc{##2}\glspostdescription}%
7307 }
7308 }
7309 {}

```

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.

```

7310 \ifcsdef{@glsstyle@long3col}
7311 {%
7312   \renewglossarystyle{long3col}{%
7313     \renewenvironment{theglossary}%
7314       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
7315       {\end{longtable}}%
7316     \renewcommand*{\glossaryheader}{}%
7317     \renewcommand*{\glsgroupheading}[1]{}%
7318     \renewcommand{\glossentry}[2]{%
7319       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7320       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline

```

```

7321 }%
7322 \renewcommand{\subglossentry}[3]{%
7323     &
7324     \glssubentryitem{##2}%
7325     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7326     ##3\tabularnewline
7327 }%
7328 \renewcommand*\{ \glsgroupskip }{%
7329     \ifglsgnogroupskip\else & &\tabularnewline\fi}%
7330 }
7331 }
7332 {}

```

Four column style:

```

7333 \ifcsdef{@glsstyle@long4col}
7334 {%
7335     \renewglossarystyle{long4col}{%
7336         \renewenvironment{theglossary}%
7337             {\begin{longtable}{llll}}%
7338             {\end{longtable}}%
7339         \renewcommand*\{ \glossaryheader }{%
7340             \renewcommand*\{ \glsgroupheading }[1]{%
7341                 \renewcommand{\glossentry}[2]{%
7342                     \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7343                     \glossentrydesc{##1}\glspostdescription &
7344                     \glossentrysymbol{##1} &
7345                     ##2\tabularnewline
7346                 }%
7347                 \renewcommand{\subglossentry}[3]{%
7348                     &
7349                     \glssubentryitem{##2}%
7350                     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7351                     \glossentrysymbol{##2} & ##3\tabularnewline
7352                 }%
7353                 \renewcommand*\{ \glsgroupskip }{%
7354                     \ifglsgnogroupskip\else & &\tabularnewline\fi}%
7355             }
7356         }
7357     }

```

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.

```

7358 \ifcsdef{@glsstyle@longragged3col}
7359 {%
7360     \renewglossarystyle{longragged3col}{%

```

```

7361 \renewenvironment{theglossary}%
7362   {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}%
7363     >{\raggedright}p{\glspagelistwidth}}}%
7364   {\end{longtable}}}%
7365 \renewcommand*{\glossaryheader}{}%
7366 \renewcommand*{\glsgroupheading}[1]{}%
7367 \renewcommand{\glossentry}[2]{%
7368   \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7369   \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
7370 }%
7371 \renewcommand{\subglossentry}[3]{%
7372   &
7373   \glssubentryitem{##2}%
7374   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7375   ##3\tabularnewline
7376 }%
7377 \renewcommand*{\glsgroupskip}{%
7378   \ifglsgnogroupskip\else & &\tabularnewline\fi}%
7379 }
7380 }
7381 {}

```

Four column style:

```

7382 \ifcsdef{@glstyle@altlongragged4col}
7383 {%
7384   \renewglossarystyle{altlongragged4col}{%
7385     \renewenvironment{theglossary}%
7386       {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}l%
7387         >{\raggedright}p{\glspagelistwidth}}}%
7388       {\end{longtable}}}%
7389     \renewcommand*{\glossaryheader}{}%
7390     \renewcommand*{\glsgroupheading}[1]{}%
7391     \renewcommand{\glossentry}[2]{%
7392       \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7393       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
7394       ##2\tabularnewline
7395     }%
7396     \renewcommand{\subglossentry}[3]{%
7397       &
7398       \glssubentryitem{##2}%
7399       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7400       \glossentrysymbol{##2} & ##3\tabularnewline
7401     }%
7402     \renewcommand*{\glsgroupskip}{%
7403       \ifglsgnogroupskip\else & & &\tabularnewline\fi}%
7404   }
7405 }
7406 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment, but not the two column styles.

```

7407 \ifcsdef{@glsstyle@super3col}
7408 {%
7409   \renewglossarystyle{super3col}{%
7410     \renewenvironment{theglossary}%
7411       {\tablehead{}\tabletail{}}%
7412       \begin{supertabular}{\lp{\glsdescwidth}p{\glspagelistwidth}}}%
7413       {\end{supertabular}}}%
7414   \renewcommand*{\glossaryheader}{}%
7415   \renewcommand*{\glsgroupheading}[1]{}%
7416   \renewcommand{\glossentry}[2]{%
7417     \glstryitem{##1}\glstarget{##1}{\glsentryname{##1}} &
7418     \glsentrydesc{##1}\glspostdescription & ##2\tabularnewline
7419   }%
7420   \renewcommand{\subglossentry}[3]{%
7421     &
7422     \glssubentryitem{##2}%
7423     \glstarget{##2}{\strut}\glsentrydesc{##2}\glspostdescription &
7424     ##3\tabularnewline
7425   }%
7426   \renewcommand*{\glsgroupskip}{%
7427     \ifglsnogroupskip\else & &\tabularnewline\fi}%
7428 }
7429 }
7430 {}

```

Four column styles:

```

7431 \ifcsdef{@glsstyle@super4col}
7432 {%
7433   \renewglossarystyle{super4col}{%
7434     \renewenvironment{theglossary}%
7435       {\tablehead{}\tabletail{}}%
7436       \begin{supertabular}{\llll}}}%
7437       {\end{supertabular}}}%
7438   \renewcommand*{\glossaryheader}{}%
7439   \renewcommand*{\glsgroupheading}[1]{}%
7440   \renewcommand{\glossentry}[2]{%
7441     \glstryitem{##1}\glstarget{##1}{\glsentryname{##1}} &
7442     \glsentrydesc{##1}\glspostdescription &
7443     \glsentrysymbol{##1} & ##2\tabularnewline
7444   }%
7445   \renewcommand{\subglossentry}[3]{%
7446     &
7447     \glssubentryitem{##2}%
7448     \glstarget{##2}{\strut}\glsentrydesc{##2}\glspostdescription &
7449     \glsentrysymbol{##2} & ##3\tabularnewline
7450   }%
7451   \renewcommand*{\glsgroupskip}{%

```

```

7452     \ifglsnogroupskip\else & & \tabularnewline\fi}%
7453 }
7454 }
7455 {}

```

2.6 Super Ragged Styles

The three and four column styles require adjustment, but not the two column styles.

```

7456 \ifcsdef{@glsstyle@superragged3col}
7457 {%
7458   \renewglossarystyle{superragged3col}{%
7459     \renewenvironment{theglossary}%
7460       {\tablehead{}\tabletail}%
7461       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%
7462         >{\raggedright}p{\glspagelistwidth}}}%
7463       {\end{supertabular}}}%
7464   \renewcommand*{\glossaryheader}{}%
7465   \renewcommand*{\glsgroupheading}[1]{}%
7466   \renewcommand{\glossentry}[2]{%
7467     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7468     \glossentrydesc{##1}\glspostdescription &
7469     ##2\tabularnewline
7470   }%
7471   \renewcommand{\subglossentry}[3]{%
7472     &
7473     \glssubentryitem{##2}%
7474     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7475     ##3\tabularnewline
7476   }%
7477   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
7478     &\tabularnewline\fi}%
7479 }
7480 }
7481 {}

```

Four columns:

```

7482 \ifcsdef{@glsstyle@altsuperragged4col}
7483 {%
7484   \renewglossarystyle{altsuperragged4col}{%
7485     \renewenvironment{theglossary}%
7486       {\tablehead{}\tabletail}%
7487       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}l%
7488         >{\raggedright}p{\glspagelistwidth}}}%
7489       {\end{supertabular}}}%
7490   \renewcommand*{\glossaryheader}{}%
7491   \renewcommand{\glossentry}[2]{%
7492     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7493     \glossentrydesc{##1}\glspostdescription &
7494     \glossentrysymbol{##1} & ##2\tabularnewline

```



```

7495 }%
7496 \renewcommand{\subglossentry}[3]{%
7497     &
7498     \glssubentryitem{##2}%
7499     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
7500     \glossentrysymbol{##2} & ##3\tabularnewline
7501 }%
7502 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & &
7503     &\tabularnewline\fi}%
7504 }
7505 }
7506 {}

```

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`.

```

7507 \ifdef{\@glsstyle@inline}
7508 {%
7509     \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
7510     Just use \glsxtrpostdescription instead of \glspostdescription.
7511     \renewcommand*{\glsinlinedescformat}[3]{%
7512         \space#1\glsxtrpostdescription}
7513     \renewcommand*{\glsinlinesubdescformat}[3]{%
7514         #1\glsxtrpostdescription}
7515 }

```

2.8 Tree Styles

The `alttree` style is redefined to make it easier to made minor adjustments.

```

7516 \ifdef{\@glsstyle@alttree}
7517 {%

```

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.

```

7518 \newcommand{\glxtraltrtreeSymbolDescLocation}[2]{%
7519     {%
7520         \let\par\glsxtrAltTreePar

```

```

7521      \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
7522      \glossentrydesc{#1}\glspostdescription \space #2\par
7523  }%
7524  }

```

`\trAltTreeIndent` Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

7525  \newlength\glxstrAltTreeIndent

```

`\lsxtrAltTreePar` Multi-paragraph descriptions need to keep the hanging indent.

```

7526  \newcommand{\glxstrAltTreePar}{%
7527    @@par
7528    \glxstrAltTreeSetHangIndent
7529    \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
7530  }

```

`\symbolDescLocation` `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

7531  \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
7532    \glxstralttreeSymbolDescLocation{#2}{#3}%
7533  }

```

`\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.

```

7534  \newlength\glxstrtreeindent

```

`\glxstralttreeInit` User-level initialisation for the alttree style.

```

7535  \newcommand*{\glxstralttreeInit}{%
7536    \settowidth{\glxstrtreeindent}{\glstreenamfmt{\glsgetwidestname\space}}%
7537    \glxstrAltTreeIndent=\parindent
7538  }

```

`\eglissetwidest` The original `\glissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

7539  \newcommand*{\eglissetwidest}[2][0]{%
7540    \protected@csedef{@glswidestname\romannumeral#1}{#2}%
7541  }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```

7542  \newcommand*{\xglissetwidest}[2][0]{%
7543    \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
7544  }

```

`\lsgetwidestname` Provide a user-level macro to obtain the widest top-level name.

```

7545  \newcommand*{\lsgetwidestname}{\@glswidestname}

```

`\getwidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```
7546 \newcommand*\glsgetwidestsubname}[1]{%
7547   \ifcsundef{@glswidestname\romannumeral#1}%
7548     {\@glswidestname}%
7549     {\csuse{@glswidestname\romannumeral#1}}%
7550 }
```

`\findTopLevelName` CamelCase is easier for long command names. Provide a CamelCase synonym of `\glsfindwidesttoplevelname`

```
7551 \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.

```
7552 \newrobustcmd*\glsFindWidestUsedTopLevelName}[1][\@glo@types]{%
7553   \dimen@=0pt\relax
7554   \gls@tmplen=0pt\relax
7555   \foralllglossaries[#1]{\@gls@type}%
7556   {%
7557     \forglsentries[\@gls@type]{\@glo@label}%
7558     {%
7559       \ifglsused{\@glo@label}%
7560       {%
7561         \ifglshasparent{\@glo@label}%
7562         {}%
7563         {%
7564           \settowidth{\dimen@}%
7565             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
7566           \ifdim\dimen@>\gls@tmplen
7567             \gls@tmplen=\dimen@
7568             \eglssetwidest{\glsentryname{\@glo@label}}%
7569           \fi
7570         }%
7571       }%
7572     }%
7573   }%
7574 }%
7575 }
```

`\destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```
7576 \newrobustcmd*\glsFindWidestUsedAnyName}[1][\@glo@types]{%
7577   \dimen@=0pt\relax
7578   \gls@tmplen=0pt\relax
7579   \foralllglossaries[#1]{\@gls@type}%
7580   {%
7581     \forglsentries[\@gls@type]{\@glo@label}%
7582     {%
```

```

7583     \ifglused{\@glo@label}%
7584     {%
7585         \settowidth{\dimen@}%
7586         {\glstreenamefmt{\glentryname{\@glo@label}}}%
7587         \ifdim\dimen@>\gls@tmplen
7588             \gls@tmplen=\dimen@
7589             \eglssetwidest{\glentryname{\@glo@label}}%
7590         \fi
7591     }%
7592     {%
7593     }%
7594 }%
7595 }

```

FindWidestAnyName Like the above but doesn't check if the entry has been used.

```

7596 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
7597     \dimen@=0pt\relax
7598     \gls@tmplen=0pt\relax
7599     \forallglossaries[#1]{\@gls@type}%
7600     {%
7601         \forallglentries[\@gls@type]{\@glo@label}%
7602         {%
7603             \settowidth{\dimen@}%
7604             {\glstreenamefmt{\glentryname{\@glo@label}}}%
7605             \ifdim\dimen@>\gls@tmplen
7606                 \gls@tmplen=\dimen@
7607                 \eglssetwidest{\glentryname{\@glo@label}}%
7608             \fi
7609         }%
7610     }%
7611 }

```

FindUsedLevelTwo This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

7612 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
7613     \dimen@=0pt\relax
7614     \dimen@i=0pt\relax
7615     \dimen@ii=0pt\relax
7616     \forallglossaries[#1]{\@gls@type}%
7617     {%
7618         \forallglentries[\@gls@type]{\@glo@label}%
7619         {%
7620             \ifglused{\@glo@label}%
7621             {%
7622                 \ifglshasparent{\@glo@label}%
7623                 {%
7624                     \edef\@glo@parent{\csuse{glo@glsetoklabel}{\@glo@label}@parent}}%
7625                     \ifglshasparent{\@glo@parent}%
7626                 }%

```

```

7627         \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
7628         \ifglshasparent{\@glo@parent}%
7629         {%
7630         {%
7631             \settowidth{\gls@tmplen}%
7632             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7633             \ifdim\gls@tmplen>\dimen@ii
7634             \dimen@ii=\gls@tmplen
7635             \eglssetwidest[2]{\glsentryname{\@glo@label}}%
7636             \fi
7637         }%
7638     }%
7639     {%
7640         \settowidth{\gls@tmplen}%
7641         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7642         \ifdim\gls@tmplen>\dimen@i
7643         \dimen@i=\gls@tmplen
7644         \eglssetwidest[1]{\glsentryname{\@glo@label}}%
7645         \fi
7646     }%
7647 }%
7648 {%
7649     \settowidth{\gls@tmplen}%
7650     {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7651     \ifdim\gls@tmplen>\dimen@
7652     \dimen@=\gls@tmplen
7653     \eglssetwidest{\glsentryname{\@glo@label}}%
7654     \fi
7655 }%
7656 }%
7657 {}%
7658 }%
7659 }%
7660 }

```

dWidestLevelTwo This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

7661 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
7662     \dimen@=0pt\relax
7663     \dimen@i=0pt\relax
7664     \dimen@ii=0pt\relax
7665     \foralllglossaries[#1]{\@gls@type}%
7666     {%
7667         \forglsentries[\@gls@type]{\@glo@label}%
7668         {%
7669             \ifglshasparent{\@glo@label}%
7670             {%
7671                 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
7672                 \ifglshasparent{\@glo@parent}%
7673                 {%

```

```

7674      \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
7675      \ifglshasparent{\@glo@parent}%
7676      {}%
7677      {%
7678        \settowidth{\gls@tmplen}%
7679        {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7680        \ifdim\gls@tmplen>\dimen@ii
7681          \dimen@ii=\gls@tmplen
7682          \eglssetwidest[2]{\glsentryname{\@glo@label}}%
7683        \fi
7684      }%
7685    }%
7686    {%
7687      \settowidth{\gls@tmplen}%
7688      {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7689      \ifdim\gls@tmplen>\dimen@i
7690        \dimen@i=\gls@tmplen
7691        \eglssetwidest[1]{\glsentryname{\@glo@label}}%
7692      \fi
7693    }%
7694  }%
7695  {%
7696    \settowidth{\gls@tmplen}%
7697    {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7698    \ifdim\gls@tmplen>\dimen@
7699      \dimen@=\gls@tmplen
7700      \eglssetwidest{\glsentryname{\@glo@label}}%
7701    \fi
7702  }%
7703 }%
7704 }%
7705 }

```

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.

```

7706 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
7707   \dimen@=0pt\relax
7708   \gls@tmplen=0pt\relax
7709   #2=0pt\relax
7710   \forallglossaries[#1]{\@gls@type}%
7711   {%
7712     \forglsentries[\@gls@type]{\@glo@label}%
7713     {%
7714       \ifglused{\@glo@label}%
7715       {%
7716         \settowidth{\dimen@}%
7717         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7718         \ifdim\dimen@>\gls@tmplen
7719           \gls@tmplen=\dimen@

```

```

7720         \eglssetwidest{\glsentryname{\@glo@label}}%
7721         \fi
7722         \settowidth{\dimen@}%
7723         {\glsentrysymbol{\@glo@label}}%
7724         \ifdim\dimen@>#2\relax
7725             #2=\dimen@
7726         \fi
7727     }%
7728     {}%
7729 }%
7730 }%
7731 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

7732 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
7733     \dimen@=0pt\relax
7734     \gls@tmplen=0pt\relax
7735     #2=0pt\relax
7736     \forallglossaries[#1]{\@gls@type}%
7737     {%
7738         \forglsentries[\@gls@type]{\@glo@label}%
7739         {%
7740             \settowidth{\dimen@}%
7741             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
7742             \ifdim\dimen@>\gls@tmplen
7743                 \gls@tmplen=\dimen@
7744                 \eglssetwidest{\glsentryname{\@glo@label}}%
7745             \fi
7746             \settowidth{\dimen@}%
7747             {\glsentrysymbol{\@glo@label}}%
7748             \ifdim\dimen@>#2\relax
7749                 #2=\dimen@
7750             \fi
7751         }%
7752     }%
7753 }

```

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.

```

7754 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
7755     \dimen@=0pt\relax
7756     \gls@tmplen=0pt\relax
7757     #2=0pt\relax
7758     #3=0pt\relax
7759     \forallglossaries[#1]{\@gls@type}%
7760     {%
7761         \forglsentries[\@gls@type]{\@glo@label}%

```

```

7762      {%
7763          \ifglsused{\@glo@label}%
7764      {%
7765          \settowidth{\dimen@}%
7766              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7767          \ifdim\dimen@>\gls@tmplen
7768              \gls@tmplen=\dimen@
7769              \eglssetwidest{\glsentryname{\@glo@label}}%
7770          \fi
7771          \settowidth{\dimen@}%
7772              {\glsentrysymbol{\@glo@label}}%
7773          \ifdim\dimen@>#2\relax
7774              #2=\dimen@
7775          \fi
7776          \settowidth{\dimen@}%
7777              {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
7778          \ifdim\dimen@>#3\relax
7779              #3=\dimen@
7780          \fi
7781      }%
7782  }%
7783 }%
7784 }%
7785 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

7786 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
7787     \dimen@=0pt\relax
7788     \gls@tmplen=0pt\relax
7789     #2=0pt\relax
7790     #3=0pt\relax
7791     \forallglossaries[#1]{\@gls@type}%
7792     {%
7793         \forglsentries[\@gls@type]{\@glo@label}%
7794     {%
7795         \settowidth{\dimen@}%
7796             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
7797         \ifdim\dimen@>\gls@tmplen
7798             \gls@tmplen=\dimen@
7799             \eglssetwidest{\glsentryname{\@glo@label}}%
7800         \fi
7801         \settowidth{\dimen@}%
7802             {\glsentrysymbol{\@glo@label}}%
7803         \ifdim\dimen@>#2\relax
7804             #2=\dimen@
7805         \fi
7806         \settowidth{\dimen@}%
7807             {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
7808         \ifdim\dimen@>#3\relax

```



```

7809         #3=\dimen@
7810     \fi
7811 }%
7812 }%
7813 }

```

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.

```

7814 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
7815     \dimen@=0pt\relax
7816     \gls@tmplen=0pt\relax
7817     #2=0pt\relax
7818     \forallglossaries[#1]{\@gls@type}%
7819     {%
7820         \forglseentries[\@gls@type]{\@glo@label}%
7821         {%
7822             \ifglssused{\@glo@label}%
7823             {%
7824                 \settowidth{\dimen@}%
7825                 {\glstreenamefmt{\glseentryname{\@glo@label}}}%
7826                 \ifdim\dimen@>\gls@tmplen
7827                     \gls@tmplen=\dimen@
7828                     \eglissetwidest{\glseentryname{\@glo@label}}%
7829                 \fi
7830                 \settowidth{\dimen@}%
7831                 {\GlsXtrFormatLocationList{\glseentrynumberlist{\@glo@label}}}%
7832                 \ifdim\dimen@>#2\relax
7833                     #2=\dimen@
7834                 \fi
7835             }%
7836         }%
7837     }%
7838 }%
7839 }

```

AnyNameLocation Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

7840 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
7841     \dimen@=0pt\relax
7842     \gls@tmplen=0pt\relax
7843     #2=0pt\relax
7844     \forallglossaries[#1]{\@gls@type}%
7845     {%
7846         \forglseentries[\@gls@type]{\@glo@label}%
7847         {%
7848             \settowidth{\dimen@}%
7849             {\glstreenamefmt{\glseentryname{\@glo@label}}}%
7850             \ifdim\dimen@>\gls@tmplen
7851                 \gls@tmplen=\dimen@

```

```

7852         \eglssetwidest{\glstryname{\@glo@label}}}%
7853     \fi
7854     \settowidth{\dimen@}%
7855         {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
7856     \ifdim\dimen@>#2\relax
7857         #2=\dimen@
7858     \fi
7859 }%
7860 }%
7861 }

```

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`.

```

7862 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
7863     \glstreeindent=\glsxtrtreetopindent\relax
7864 }

```

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.

```

7865 \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
7866     \ifcsundef{@glswidestname\romannumeral#1}%
7867     {%
7868         \settowidth{#3}{\glstreenamfmt{\@glswidestname\space}}%
7869     }%
7870     {%
7871         \settowidth{#3}{\glstreenamfmt{%
7872             \csname @glswidestname\romannumeral#1\endcsname\space}}%
7873     }%
7874 }

```

treeSetHangIndent Set `\hangindent` for top-level entries:

```

7875 \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

treeSetSubHangIndent Set `\hangindent` for sub-entries:

```

7876 \newcommand*{\glsxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine `alttree`:

```

7877 \renewglossarystyle{alttree}{%
7878     \renewenvironment{theglossary}%
7879     {%
7880         \glsxtralttreeInit

```

```

7881     \def\@gls@prevlevel{-1}%
7882     \mbox{}\par}%
7883     {\par}%
7884 \renewcommand*\glossaryheader{}%
7885 \renewcommand*\glsgroupheading[1]{}%
7886 \renewcommand\glossentry[2]{%
7887     \ifnum\@gls@prevlevel=0\relax
7888     \else
7889         \glstrComputeTreeIndent{##1}%
7890     \fi
7891     \parindent\glstreeindent
7892     \glstrAltTreeSetHangIndent
7893     \makebox[Opt][r]%
7894     {%
7895         \glstreenamebox{\glstreeindent}%
7896         {%
7897             \glstryitem{##1}%
7898             \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
7899         }%
7900     }%
7901     \glstralttreeSymbolDescLocation{##1}{##2}%
7902     \def\@gls@prevlevel{0}%
7903 }
7904 \renewcommand\subglossentry[3]{%
7905     \ifnum##1=1\relax
7906         \glssubentryitem{##2}%
7907     \fi
7908     \ifnum\@gls@prevlevel=##1\relax
7909     \else
7910         \glstrComputeTreeSubIndent{##1}{##2}{\@gls@tmplen}%
7911         \ifnum\@gls@prevlevel<##1\relax
7912             \setlength\glstreeindent\@gls@tmplen
7913             \addtolength\glstreeindent\parindent
7914             \parindent\glstreeindent
7915         \else
7916             \ifnum\@gls@prevlevel=0\relax
7917                 \glstrComputeTreeIndent{##2}%
7918             \else
7919                 \glstrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
7920             \fi
7921             \addtolength\parindent{-\glstreeindent}%
7922             \setlength\glstreeindent\parindent
7923         \fi
7924     \fi
7925     \glstrAltTreeSetSubHangIndent{##1}%
7926     \makebox[Opt][r]{\glstreenamebox{\@gls@tmplen}{%
7927         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
7928     \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
7929     \def\@gls@prevlevel{##1}%

```

```

7930     }%
7931     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7932   }
7933 }%
7934 {%

```

Assume the style isn't required if it hasn't already been defined.

```

7935 }

```

Reset the default style

```

7936 \ifx\@glossary@default@style\relax
7937 \else
7938   \setglossarystyle{\@glxtr@current@style}
7939 \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	46
General: Initial experimental release	4	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	47
\Glsfmtshort: new	204	\@Glstext@: added redefinition	42
\glsfmtshort: new	203	\@Glsuseri@: added redefinition	47
\Glsfmtshortpl: new	204	\@Glsuserii@: added redefinition	47
\glsfmtshortpl: new	203	\@Glsuseriii@: added redefinition	48
short: switched inline full form to short		\@Glsuseriv@: added redefinition	48
(long)	166	\@Glsuserv@: added redefinition	48
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	49
\@ACRlong: added redefinition	52	\@acrlong: added redefinition	51
\@ACRlongpl: added redefinition	53	\@acrlongpl: added redefinition	52
\@ACRshort: added redefinition	50	\@acrshort: added redefinition	49
\@ACRshortpl: added redefinition	51	\@acrshortpl: added redefinition	50
\@Acrlong: added redefinition	51	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	52	argument	39
\@Acrshort: added redefinition	49	\@glsdescplural@: added redefinition	45
\@Acrshortpl: added redefinition	50	\@glsfirst@: added redefinition	42
\@GLSdesc@: added redefinition	45	\@glsfirstplural@: added redefinition	44
\@GLSdescplural@: added redefinition	46	\@glsplural@: added redefinition	43
\@GLSfirst@: added redefinition	43	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	44	redefinition	46
\@GLSname@: added redefinition	45	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	44	new	97
\@GLSSymbol@: added redefinition	46	\@glsxtr@field@linkdefs: new	41
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	138
redefinition	47	\@print@glossary: added redefinition	94
\@GLStext@: added redefinition	42	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	47	\abbrvdefaultfont	142
\@GLSuserii@: added redefinition	48	\glsaccessdesc: new	108
\@GLSuseriii@: added redefinition	48	\glsaccessdescplural: new	109
\@GLSuseriv@: added redefinition	48	\glsaccessfirst: new	106
\@GLSuserv@: added redefinition	48	\glsaccessfirstplural: new	106
\@GLSuservi@: added redefinition	49	\Glsaccesslong: new	111
\@Glsdesc@: added redefinition	45	\glsaccesslong: new	110
\@Glsdescplural@: added redefinition	46	\glsaccessname: new	104
\@Glsfirst@: added redefinition	43	\glsaccessplural: new	105
\@Glsfirstplural@: added redefinition	44	\Glsaccessshort: new	110
\@Glsname@: added redefinition	45	\glsaccessshort: new	109
\@Glsplural@: added redefinition	43	\Glsaccessshortpl: new	110

\glsaccesssshortpl: new	110	\cGLSpl: new	75
\glsaccessssymbol: new	107	\cGLSpl@: new	75
\glsaccessssymbolplural: new	108	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	105	new	70
\glstentryfmt: added check for short ..	39	\cGLS: new	75
\glslongpltok: new	138	\cGLSformat: new	75
\glsshortpltok: new	138	\cGLSpl: new	75
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	75
name in \setkeys	139	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	10
for plural	135	\glsenableentrycount: new	71
\GLSxtrlongpl: new	152	\glsfirstabbrvdefaultfont: new ..	142
\Glsxtrlongpl: new	151	\glsfirstlongdefaultfont: new ...	142
\glsxtrlongpl: new	151	\Glsfmtfirst: new	206
\glsxtrNoGlossaryWarning: new	14	\glsfmtfirst: new	205
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	206
new	135	\glsfmtfirstpl: new	206
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	205
new	135	\glsfmtplural: new	205
\glsxtrpostlinkendsentence: new ..	134	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	150	\Glsxtrtitleshort	204
\Glsxtrshortpl: new	149	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	203
\Glsxtrshort	204	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl	203
\glsxtrshort	203	\Glsfmttext: new	205
\Glsfmtshortpl: changed to use		\glsfmttext: new	204
\glsxtrshortpl	204	\glshasattribute: new	116
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	115
\glsxtrshortpl	203	\GlsXtrEnableEntryCounting: new ..	70
\glsxtrifemptyglossary: new	16	\glsxtrifcounttrigger: new	73
\glsxtrnewnumber: added extra		\glsxtrscfont: new	170
argument	119	\glsxtrscsuffix: new	170
\glsxtrnewsymbol: added extra		\glsxtrsmfont: new	174
argument	119	\glsxtrsmsuffix: new	174
\MakeAcronymsAbbreviations: set the		short-em: new	181
default type to \acronymtype	84	short-em-desc: new	181
\newterm: fixed name argument	118	short-em-footnote: new	183
0.5 (2015-12-07)		short-em-long: new	179
\cGLS: new	75	short-em-long-desc: new	180
\cGLS@: new	75	short-em-postfootnote: new	183

short-sc-footnote: new	173	\Glsxtrheadtext: now uses headuc	
short-sc-postfootnote: new	174	attribute	197
short-sm: new	175	\glxtrheadtext: now uses headuc	
short-sm-desc: new	176	attribute	196
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	182	if set	158
long-noshort-sm: new	176	long-short-desc: switch off regular	
long-noshort-sm-desc: new	176	attribute if set	159
long-short-em: new	178	footnote: switch off regular attribute if	
long-short-em-desc: new	178	set	162
long-short-sm: new	174	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	45
0.5.1 (2015-12-07)		\@GLSdescplural@: added accessibility	
\@glxtr@doaccsupp: new	13	support	46
General: removed \ifglxtruseuhead	195	\@GLSfirst@: added accessibility	
\Glsaccessdesc: new	108	support	43
\Glsaccessdescplural: new	109	\@GLSfirstplural@: added accessibility	
\Glsaccessfirst: new	106	support	44
\Glsaccessfirstplural: new	107	\@GLSname@: added accessibility support	45
\Glsaccessname: new	104	\@GLSplural@: added accessibility	
\Glsaccessplural: new	105	support	44
\Glsaccesssymbol: new	107	\@GLSsymbol@: added accessibility	
\Glsaccesssymbolplural: new	108	support	46
\Glsxtrheadfirst: now uses headuc		\@GLSsymbolplural@: added	
attribute	198	accessibility support	47
\glxtrheadfirst: now uses headuc		\@GLStext@: added accessibility support	42
attribute	198	\@GLSdesc@: added accessibility support	45
\Glsxtrheadfirstplural: now uses		\@GLSdescplural@: added accessibility	
headuc attribute	199	support	46
\glxtrheadfirstplural: now uses		\@GLSfirst@: added accessibility	
headuc attribute	199	support	43
\Glsxtrheadplural: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	198	support	44
\glxtrheadplural: now uses headuc		\@GLSname@: add accessibility support ..	45
attribute	197	\@GLsplural@: added accessibility	
\Glsxtrheadshort: now uses headuc		support	43
attribute	196	\@GLSsymbol@: added accessibility	
\glxtrheadshort: now uses headuc		support	46
attribute	195	\@GLSsymbolplural@: added	
\Glsxtrheadshortpl: now uses headuc		accessibility support	47
attribute	196	\@GLStext@: added accessibility support	42
\glxtrheadshortpl: now uses headuc		\@GLSdesc@: added accessibility support	45
attribute	195		

\@glsdescplural@: added accessibility support	45	\glsxtrnewabbrevpresetkeyhook: new	140
\@glsfirst@: added accessibility support	42	\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	43	\setabbreviationstyle: added check for post-definition style switch	155
\@glsymbol@: added accessibility support	46	0.5.3 (2015-12-09)	
\@glsymbolplural@: added accessibility support	46	\@glsxtr@autoindex@at: new	128
\@glstext@: added accessibility support	42	\@glsxtr@autoindex@encap: new ...	129
\@glsxtr@activate@initialtagging: new	133	\@glsxtr@autoindex@esc: new	129
\@glsxtr@do@titlecaps@warn: new	132	\@glsxtr@autoindex@level: new ...	129
\@glsxtr@tag: new	133	\@glsxtr@autoindex@setname: new	127
General: fixed typo in glossaries-accsupp and tidied up code to use just one		\@glsxtr@doabbreviationsdef: new	11
\@ifpackageloaded	104	General: removed	
removed \glsxtrabbrvfmt	152	\GlsXtrNoGlsWarningNoAutoMakeMain	96
\glossaryentrynumbers: added	36	\glsdescwidth: added	35
\Glossentrydesc: added	130	\glspagelistwidth: added	36
\Glossentryname: added	125	\glsxtrdoautoindexname: new	127
\Glossentrysymbol: added	131	\glsxtrpostnamehook: new	126
\glossentrysymbol: added	131	\if@glsxtr@format@override: new	126
\GLSaccessdesc: new	109, 113	\ProvidesGlossariesExtraLang: new	209
\GLSaccessdescplural: new ...	109, 114	\RequireGlossariesExtraLang: new	209
\GLSaccessfirst: new	106, 112	0.5.4 (2015-12-15)	
\GLSaccessfirstplural: new ..	107, 113	\@newglossaryentry@defunitcounters: new	76
\GLSaccesslong: new	111, 114	\@GLSxtr@p@acrlong@: new	64
\GLSaccesslongpl: new	111, 114	\@GLSxtr@p@acrlongpl@: new	64
\Glsaccesslongpl: new	111	\@GLSxtr@p@acrshort@: new	63
\glsaccesslongpl: new	111	\@GLSxtr@p@acrshortpl@: new	63
\GLSaccessname: new	104, 112	\@GLSxtr@p@long@: new	63
\GLSaccessplural: new	106, 112	\@GLSxtr@p@longpl@: new	63
\GLSaccessshort: new	110, 114	\@GLSxtr@p@plural@: new	62
\GLSaccessshortpl: new	110, 114	\@GLSxtr@p@short@: new	62
\GLSaccesssymbol: new	107, 113	\@GLSxtr@p@shortpl@: new	62
\GLSaccesssymbolplural: new	108, 113	\@GLSxtr@p@text@: new	61
\GLSaccessstext: new	105, 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	131	\@Glsxtr@p@acrshort@: new	63
\glsxtrfieldtitlecase: new	120	\@Glsxtr@p@acrshortpl@: new	63
\GlsXtrFormatLocationList: new ...	36	\@Glsxtr@p@long@: new	63
		\@Glsxtr@p@longpl@: new	63
		\@Glsxtr@p@plural@: new	62
		\@Glsxtr@p@short@: new	62
		\@Glsxtr@p@shortpl@: new	62
		\@Glsxtr@p@text@: new	61

\@Glsxtrpl: new	33	\glxtr: new	32
\@alt@glshyp@opt: new	59	\glxtrcat: new	32
\@glshalt@hyp@opt: new	58	\glxtrdowrglossaryhook: new	58
\@glshalt@hyp@opt@char: new	59	\GlsXtrEnableEntryUnitCounting:	
\@glshalt@hyp@opt@keys: new	59	new	81
\@glshincrement@currunitcount:		\GlsXtrEnableOnTheFly: new	31
new	77	\Glsxtrpl: new	33
\@glshlocal@increment@currunitcount:		\glxtrpl: new	33
new	78	\glxtrpostlocalreset: new	70
\@glshsetdefault@glslink@opts:		\glxtrpostlocalunset: new	69
new	56	\glxtrpostreset: new	70
\@glsxtr: new	32	\glxtrpostunset: new	69
\@glsxtr@addunitcounter: new	77	\glxtrprotectlinks: new	60
\@glsxtr@currunitcount: new	78	\GlsXtrSetAltModifier: new	59
\@glsxtr@ifunitcounter: new	77	\GlsXtrSetDefaultGlsOpts: new	57
\@glsxtr@p@acrlong@: new	63	\glxtrstarflywarn: new	32
\@glsxtr@p@acrlongpl@: new	64	\GlsXtrWarning: new	34
\@glsxtr@p@acrshort@: new	63	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new	63	disables \setacronymstyle	84
\@glsxtr@p@long@: new	63	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new	63	\@glsxtr@autoindexcrossrefs: new .	10
\@glsxtr@p@plural@: new	61	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new	62	new	90
\@glsxtr@p@shortpl@: new	62	\@glsxtr@idx@entrynumberlist: new	92
\@glsxtr@p@text@: new	61	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new	78	new	90
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new	82	new	91
\@glsxtr@unitcountlist: new	76	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new	33	new	91
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new	85
empty see value if not set and added		\makeglossaries: new	85
‘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	76	\glxtrinlinefullformat and	
\cGlsplformat: added	76	added missing second argument ...	167
\glshdisablehyper: added	60	1.02 (2016-04-25)	
\glshdohyperlink: added	59	\@glsxtr@current@style: new	34
\glshdonohyperlink: added	60	\Glsfmtfull: new	208
\glshenableentryunitcount: new	78	\glsfmtfull: new	208
\glshasattribute: added check for		\Glsfmtfullpl: new	209
entry’s existence	116	\glsfmtfullpl: new	208
\glshifattribute: added check for		\Glsfmtlong: new	207
entry’s existence	116	\glsfmtlong: new	207
\glshpostlinkhook: added existence		\Glsfmtlongpl: new	207
check	134	\glsfmtlongpl: new	207
\Glsxtr: new	32	\Glsxtrheadfull: new	202

\glxtrheadfull: new	201	\@GLSplural@: set abbreviation and regular format	44
\Glsxtrheadfullpl: new	203	\@GLSsymbol@: set regular format	46
\glxtrheadfullpl: new	202	\@GLSsymbolplural@: set regular format	47
\glxtrheadlong: new	201	\@GLStext@: set abbreviation and regular format	42
\Glsxtrheadlongpl: new	201	\@GLSuseri@: set regular format	47
\glxtrheadlongpl: new	200	\@GLSuserii@: set regular format	48
\Glsxtrtitlefull: new	203	\@GLSuseriii@: set regular format	48
\glxtrtitlefull: new	202	\@GLSuseriv@: set regular format	48
\Glsxtrtitlefullpl: new	203	\@GLSuserv@: set regular format	48
\glxtrtitlefullpl: new	202	\@GLSuservi@: set regular format	49
\Glsxtrtitlelong: new	201	\@Glsdesc@: set abbreviation and regular format	45
\glxtrtitlelong: new	200	\@Glsdescplural@: set abbreviation and regular format	46
\Glsxtrtitlelongpl: new	201	\@Glsfirst@: set abbreviation and regular format	43
\glxtrtitlelongpl: new	200	\@Glsfirstplural@: set abbreviation and regular format	44
\ifglxtrinsetinside: new	158	\@Glsname@: set abbreviation and regular format	45
postfootnote: added redef of \glxtrsetupfulldefs	164	\@Glsplural@: set abbreviation and regular format	43
stylemods: new	14	\@Glsymbol@: set regular format	46
1.03 (2016-04-27)		\@Glsymbolplural@: set regular format	47
\@GLSfirstplural@: bug fix: misspelt cs name	44	\@Glstext@: set abbreviation and regular format	42
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	44	\@GLSuseri@: set regular format	47
\@Glsfirstplural@: bug fix: misspelt cs name	44	\@GLSuserii@: set regular format	47
\@Glsplural@: fixed bug \@Glsplural@ should be redefined not \@Glsplural	43	\@GLSuseriii@: set regular format	48
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	43	\@GLSuseriv@: set regular format	48
\glxtrtitlelongpl: bug fix: changed \glxtrlong to \glxtrlongpl ..	200	\@GLSuserv@: set regular format	48
\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	45
1.04 (2016-05-02)		\@glsdescplural@: set abbreviation and regular format	45
\@@glxtrpostloctag: new	38	\@glsfirst@: set abbreviation and regular format	42
\@GLSdesc@: set abbreviation and regular format	45	\@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 ..	43	\@glsplural@: set abbreviation and regular format	43
\@GLSfirstplural@: set abbreviation and regular format	44	\@glsymbol@: set regular format	46
\@GLSname@: set abbreviation and regular format	45		

\@glssymbolplural@: set regular format	46	short-em-nolong-desc: new	181
\@glstext@: set abbreviation and regular format	42	short-em-postfootnote: renamed from “postfootnote-em”	183
\@glstr@deprecated@abbrstyle: new	156	short-footnote: new	163
\@glstr@do@style: new	15	short-long-user: new	190
\@glstr@doloctag: new	38	short-long-user-desc: new	191
\@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”	173
\@glstr@postloctag: new	38	short-sc-nolong: new	172
\@glstr@preloctag: new	37	short-sc-nolong-desc: new	172
\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	122	short-sm-nolong: new	176
moved post name hook inside condition	124	short-sm-nolong-desc: new	176
\glsabbrvmfont: new	178	short-sm-postfootnote: renamed from “postfootnote-sm”	177
\glsabbrvuserfont: new	184	\letabbreviationstyle: new	156
\glsfirstabbrvmfont: new	178	\newabbreviationstyle: bug fix: corrected test for existence	155
\glsfirstabbrvuserfont: new	184	long-em-noshort-em: new	182
\glsfirstlongemfont: new	178	long-em-noshort-em-desc: new	183
\glsfirstlonguserfont: new	185	long-em-short-em: new	178
\glsifnotregularcategory: new	117	long-em-short-em-desc: new	179
\glslongdefaultfont: new	142	long-noshort: new	170
\glslongemfont: new	178	long-noshort-desc: new	170
\glslongfont: new	142	long-noshort-em: renamed from “long-em”	182
\glslonguserfont: new	184	long-noshort-em-desc: renamed from “long-desc-em”	182
\glstrassignfieldfont: new	41	long-noshort-sc: renamed from “long-sc”	172
\GlsXtrEnablePreLocationTag: new	37	long-noshort-sc-desc: renamed from “long-desc-sc”	173
\glstrfirstscfont: new	170	long-noshort-sm: renamed from “long-sm”	176
\glstrfirstsmfont: new	174	long-noshort-sm-desc: renamed from \long-desc-sm	176
\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	43
\glsFindWidestAnyNameSymbol: new	223	\@GLSfirstplural@: added check for nohyperfirst attribute	44
\glsFindWidestAnyNameSymbolLocation: new	224	\@GLSxtrp: new	65
\glsFindWidestLevelTwo: new	221	\@Glsfirst@: added check for nohyperfirst attribute	43
\glsFindWidestUsedAnyName: new	219	\@Glsfirstplural@: added check for nohyperfirst attribute	44
\glsFindWidestUsedAnyNameLocation: new	225	\@Glsxtrp: new	65
\glsFindWidestUsedAnyNameSymbol: new	222	\@gls@preglossaryhook: added \glossxtrsetpopts	133
\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	161	\@glsxtrnotinmark: new	193
\glsgetwidestname: new	218	\@glsxtrp: new	64
\glsgetwidestsubname: new	219	\@glsxtrp@opt: new	64
\glslongfootnotefont: new	161	\glossxtrsetpopts: new	64
\glsxtrAltTreeIndent: new	218	\glsps: new	67
\glsxtralttreeInit: new	218	\glspt: new	67
\glsxtrAltTreePar: new	218	\glsxtr@entry@p: new	65
\glsxtrAltTreeSetHangIndent: new	226	\glsxtrabbrvfootnote: new	162
\glsxtrAltTreeSetSubHangIndent: new	226	\glsxtrchecknohyperfirst: new	42
\glsxtralttreeSubSymbolDescLocation: new	218	\glsxtrfieldtitlecasecs: new	120
\glsxtralttreeSymbolDescLocation: new	217	\glsxtrifinmark: new	193
\glsxtrComputeTreeIndent: new	226	\GLSxtrp: new	68
\glsxtrComputeTreeSubIndent: new	226	\Glsxtrp: new	67
\glsxtrtreeopindent: new	218	\glsxtrp: new	66
short-em-long: fixed incorrect font used by long form	179	\glsxtrsetpopts: new	64
\xglsssetwidest: new	218	short-long-desc: added text key	161
1.06 (2016-06-18)		fixed misspelling of \glsabbrvfont in plural key	161
\@glsdoifexistSORWARN: new	10	long-short-desc: added missing text key	159
\@glsxtr@docdefval: new	9	fixed misspelling of \glsabbrvfont	159
\@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	163
docdef option changed to choice	9	switched from \glsfirstlongfont to \glsfirstlongfootnotefont	165
\glsxtr@usesee: new	29	\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse	84
\glsxtrusesee: new	28	1.08 (2016-12-13)	
\glsxtruseseeformat: new	29	\@@glsxtr@record: new	6
\if@glsxtrdocdefrestricted: new	10		
1.07 (2016-08-15)			
\@@glsxtrp: new	64		

\@GLS@: added \@glxtr@record	40	\@glxtr@redef@forglsentries: new	5
\@GLSpl@: added \@glxtr@record	40	\@glxtr@shortcutsval: new	12
\@Gls@: added \@glxtr@record	40	\@glxtr@unsrt@getgrouptitle: new	102
\@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	99
\@Gls@saveentrycounter: new	16	\@glxtr@linkprefix: new	99
\@Glsdispl: 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	99
\@glxtr@record: new	6	\@glxtr@texencoding: new	99
\@glxtr@warn@onexistsordo: new	5	\@glxtr@writefields: new	99
\@glxtr@warn@undefaction: new	5	\@GlsXtrLoadResources: new	99
\@print@unsrt@glossary: new	101	\@glxtr@resourcefile: changed	
General: added record package option	8	extension to .glstex	98
\@glxtr@add: added \@glxtr@record	41	\@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	133
new	7	\@glxtr@counterrecordhook: new	100
\@glxtr@record: new	100	\@glxtr@display@loc: new	93
\@glxtr@resource: new	99	\@glxtr@docounterrecord: new	100
\@glxtr@saveentrycounter: new	16	\@glxtr@longnewglossaryentry:	
\@glxtr@setup@record: new	7	new	23
\@glxtr@assignfieldfont: added check		\@glxtr@noop@recordcounter: new	7
for existence	41	\@glxtr@op@recordcounter: new	7
\@glxtr@resourcefile: new	98	\@glxtr@provide@storagekey: new	17
\@print@unsrt@glossaries: new	101	\@glxtr@s@longnewglossaryentry:	
\@print@unsrt@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	56
new	90	\@newglossaryentryposthook: added	
\@print@glossary: redefined to save		check for alias key	22
options	89	\@no@glxtr@indexaliased: new	57
\@glxtr@makeglossaries: new	90	\@print@unsrt@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	40	\@aptoglossarypreamble: new	21
1.11 (2017-01-19)		\@csGlsXtrLetField: new	20
\@glxtr@do@redef@forglsentries:		\@eGlsXtrSetField: new	21
new	5	\@gGlsXtrSetField: new	21
\@glxtr@noidx@do: new	103		

\glsdohyperlink: added check for alias field	60	\glsxtrindexaliased: new	57
\glsnoidxdisplayloc: added redefinition	93	\GlsXtrLetField: new	20
\glssettoctitle: added patch	25	\GlsXtrLetFieldToField: new	21
\glsxtr@counterrecord: new	100	\GlsXtrLoadResources: removed restriction on only one per document	99
\glsxtr@langtag: new	99	\glsxtrlocrangefmt: new	94
\glsxtr@newabbreviation: new	139	\glsxtrpostlongdescription: new ..	23
\glsxtr@org@newignoredglossary: Added check for existence	24	\glsxtrprovidestoragekey: new	17
\glsxtr@pluralsuffixes: new	99	\GlsXtrRecordCounter: new	100
\glsxtr@provideignoredglossary: new	25	\glsxtrresourcecount: new	99
\glsxtr@s@newignoredglossary: Added check for existence	24	\glsxtrresourcefile: added catcode change for @	98
\glsxtr@s@provideignoredglossary: new	26	\glsxtrsetaliasnoindex: new	56
\glsxtrabbrvpluralsuffix: new ...	142	\GlsXtrSetField: new	20
\glsxtralias: new	22	\glsxtrsetfieldifexists: new	20
\glsxtrcopytoglossary: new	26	\glsxtrunsrtdo: new	103
\glsxtrdeffield: new	20	\Glsxtrusefield: new	20
\glsxtrdisplayendloc: new	93	\glsxtrusefield: new	20
\glsxtrdisplayendlochook: new	94	short-postlong-user: new	188
\glsxtrdisplaysingleloc: new	93	short-postlong-user-desc: new ...	189
\glsxtrdisplaystartloc: new	93	\longnewglossaryentry: added starred version	22
\glsxtrdeffield: new	20	long-postshort-user: new	186
\glsxtrentryfmt: new	18	long-postshort-user-desc: new ...	187
\glsxtrfieldddolistloop: new	19	postdot: new	11
\glsxtrfieldforlistloop: new	19	\pretoglossarypreamble: new	22
\glsxtrfielddifylist: new	20	\print@noop@unsrtglossaryunit: new	102
\glsxtrfieldlistadd: new	19	\print@op@unsrtglossaryunit: new	102
\glsxtrfieldlistead: new	19	\printunsrtglossary: added starred form	101
\glsxtrfieldlistgadd: new	19	\printunsrtglossaryhandler: new .	102
\glsxtrfieldlistxadd: new	19	\printunsrtglossaryunit: new	7
\glsxtrfieldxifylist: new	20	\printunsrtglossaryunitsetup: new	102
\glsxtrfmt: new	18	\provideignoredglossary: new	25
\GlsXtrFmtDefaultOptions: new	18	\s@glsxtr@provide@storagekey: new	18
\GlsXtrFmtField: new	18	\s@printunsrtglossary: new	101
\glsxtrifkeydefined: new	17	\xGlsXtrSetField: new	21

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	
\.	134, 135, 217
\@	98
\@cGLS@	72, 80
\@cGLSpl@	72, 80
\@cGLspl@	72, 80
\@cgls@	72, 80
\@cglspl@	72, 73, 80
\@do@wrglossary	7, 86
\@do@wrglossary	9, 16, 41, 57
\@glo@assign@sortkey	90
\@glo@list	5
\@glo@type	101
\@gls@expand@field	18
\@glslocalreset	70
\@glslocalunset	69
\@glsreset	70
\@glsunset	69
\@glsxtr@autoindex@escspch	129, 130
\@glsxtr@checkspch	127, 128, 130
\@glsxtr@disabledflycommand	34
\@glsxtr@record	9
\@glsxtr@recordcounter	9, 100
\@glsxtrp	64, 65
\@glsxtrpostloctag	37
\@glsxtrpreloctag	37
\@newglossaryentry@defcounters	71
\@newglossaryentry@defunitcounters	78
\@par	218
\@ACRlong	61
\@ACRlongpl	61
\@ACRshort	61
\@ACRshortpl	61
\@Acrlong	61
\@Acrlongpl	61
\@Acrshort	61
\@Acrshortpl	61
\@GLS@	61, 74, 75
\@GLSdesc@	46
\@GLSpl@	61, 74, 75
\@GLSplural@	62
\@GLSsymbol@	47
\@GLStext@	61
\@GLSxtr@full	144
\@GLSxtr@fullpl	145
\@GLSxtr@p@acrlong@	61
\@GLSxtr@p@acrlongpl@	61
\@GLSxtr@p@acrshort@	61
\@GLSxtr@p@acrshortpl@	61
\@GLSxtr@p@long@	61
\@GLSxtr@p@longpl@	61
\@GLSxtr@p@plural@	61
\@GLSxtr@p@short@	61
\@GLSxtr@p@shortpl@	61
\@GLSxtr@p@text@	61
\@GLSxtrlong	61, 148, 149
\@GLSxtrlongpl	61, 152
\@GLSxtrp	68, 69
\@GLSxtrshort	61, 147
\@GLSxtrshortpl	61, 150
\@Gls@	61, 74, 75
\@Gls@acentryname	82
\@Gls@entry@field	53, 67, 68
\@Gls@entryname	82
\@GlsXtrEnableOnTheFly	31
\@Glspl@	61, 74, 75
\@Glsplural@	62
\@Glstext@	61
\@Glsxtr	32, 34
\@Glsxtr@full	143
\@Glsxtr@fullpl	145
\@Glsxtr@p@acrlong@	61
\@Glsxtr@p@acrlongpl@	61
\@Glsxtr@p@acrshort@	61

\@Glsxtr@p@acrshortpl@	61	\@glo@descplural	23
\@Glsxtr@p@long@	61	\@glo@group	8
\@Glsxtr@p@longpl@	61	\@glo@label	8, 17, 22, 28, 29, 53, 219–226
\@Glsxtr@p@plural@	61	\@glo@location	8
\@Glsxtr@p@short@	61	\@glo@loclist	8
\@Glsxtr@p@shortpl@	61	\@glo@name	127
\@Glsxtr@p@text@	61	\@glo@no@assign@sortkey	90
\@Glsxtrlong	61, 148	\@glo@parent	220–222
\@Glsxtrlongpl	61, 151	\@glo@see	22, 28, 29
\@Glsxtrp	67, 68	\@glo@sort	127
\@Glsxtrpl	33, 34	\@glo@sorttype	88, 92
\@Glsxtrshort	61, 146	\@glo@thislettergrp	103
\@Glsxtrshortpl	61, 150	\@glo@thisvalue	184
\@acrlong	61	\@glo@tmp	17, 53
\@acrlongpl	61	\@glo@type	29, 83, 85, 88–90, 92, 94, 97, 98, 101
\@acrshort	61	\@glo@types	118, 219–225
\@acrshortpl	61	\@glossary@default@style	34, 35, 89, 228
\@alt@glshyp@opt	58	\@glossarystyle	89
\@auxout	7, 38, 72, 81, 85, 86, 94, 98–101	\@glsc@	61, 73, 75
\@bibgl@restoreat	98	\@glsc@link	41
\@cGLS	75	\@glsc@actualchar	128
\@cGLS@	72, 75, 80	\@glsc@adjustmode	41
\@cGLSpl	75	\@glsc@alt@hyp@opt	59
\@cGLSpl@	72, 75, 80	\@glsc@alt@hyp@opt@char	58, 59
\@cGLspl@	72, 80	\@glsc@alt@hyp@opt@keys	59
\@cglsc@	72, 80	\@glsc@automake	88
\@cglspl@	72, 80	\@glsc@checkedmkidx	127, 128, 130
\@disable@onlypremakeg	86	\@glsc@checkmkidxchars	127
\@do@auxoutstuff	94	\@glsc@codepage	94
\@do@glscsee	22	\@glsc@counter	6, 7, 41, 57
\@do@newglossaryentry	83, 140	\@glsc@currentlettergroup	92, 101, 103
\@do@seeglossary	86	\@glsc@declareoption	4
\@empty	41, 49–53, 127, 128, 143–152	\@glsc@doautomake	88
\@end@glxtr@addunused	29, 30	\@glsc@encapchar	128
\@end@glxtr@gettype	88, 90	\@glsc@entry@count	72
\@end@glxtr@usesee	29	\@glsc@entry@field	17, 20, 53, 66–69, 71
\@endfortrue	155	\@glsc@entry@unitcount	80, 81
\@firstofone	42, 121, 122, 126, 132	\@glsc@field@font	42–49
\@firstofthree	41, 49–52, 58, 59, 143, 144, 146, 148, 149, 151	\@glsc@field@link	42–49, 54, 55
\@firstoftwo	43, 44, 46, 47, 50–53, 55, 58, 84, 136, 137, 143–145, 149–152, 193, 194	\@glsc@getgrouptitle	101
\@for	5, 14, 30, 70, 82, 85, 88, 101, 120, 131	\@glshyp@opt	54, 55, 59, 75, 143–152
\@glo@assign@sortkey	88	\@glshyp@opt@cs	58, 59
\@glo@category	76	\@glsc@increment@currcount	71
\@glo@counterprefix	7	\@glsc@increment@currunitcount	79
\@glo@countunit	76	\@glsc@keymap	8, 17, 28, 53, 99
\@glo@default@sorttype	88	\@glsc@label	7, 58, 86, 101, 155
\@glo@desc	23	\@glsc@levelchar	128
		\@glsc@link	18, 39, 41, 49–53, 143–152
		\@glsc@link@checkfirsthyper	84

<code>\@gls@link@nocheckfirsthyper</code>	<code>\@gls@target</code>	60, 90
..... 39, 49–53, 143–152	<code>\@gls@text@</code>	61
<code>\@gls@local@increment@currcount</code>	<code>\@gls@widestname</code>	218, 219, 226
71	<code>\@gls@xtr</code>	32, 34
<code>\@gls@local@increment@currunitcount</code> 79	<code>\@gls@xtr@do@wrglossary</code>	86
<code>\@gls@location</code>	<code>\@gls@xtr@abbreviationsdef</code>	11, 15, 16
103, 104	<code>\@gls@xtr@activate@initialtagging</code> ..	
<code>\@gls@loclist</code> 132, 133	
90–92, 103, 104	<code>\@gls@xtr@addunitcounter</code>	76
<code>\@gls@longpl</code>	<code>\@gls@xtr@addunusedxrefs</code>	29, 30
138–140	<code>\@gls@xtr@attrval</code> ...	121, 122, 124, 125, 127
<code>\@gls@nohyperlist</code>	<code>\@gls@xtr@autoindex@at</code>	127–129
24, 26	<code>\@gls@xtr@autoindex@doextra@esc</code>	127
<code>\@gls@noidx@do</code>	<code>\@gls@xtr@autoindex@encap</code>	127–129
92	<code>\@gls@xtr@autoindex@esc</code>	127–130
<code>\@gls@noidx@nosanitizesort</code>	<code>\@gls@xtr@autoindex@escat</code>	128, 129
88	<code>\@gls@xtr@autoindex@escencap</code> ...	128, 129
<code>\@gls@noidx@sanitizesort</code>	<code>\@gls@xtr@autoindex@esclevel</code> ...	128, 129
87	<code>\@gls@xtr@autoindex@escquote</code> ...	127, 129
<code>\@gls@noidx@loclist@finalsep</code>	<code>\@gls@xtr@autoindex@level</code>	128, 129
91	<code>\@gls@xtr@autoindex@setname</code>	127
<code>\@gls@noidx@loclist@prev</code>	<code>\@gls@xtr@autoindex@crossrefs</code> ...	9, 10, 28
91	<code>\@gls@xtr@cat</code>	70, 71, 82, 131
<code>\@gls@noidx@loclist@sep</code>	<code>\@gls@xtr@counterrecordhook</code>	7
90	<code>\@gls@xtr@csname</code>	77–80
<code>\@gls@noref@warn</code>	<code>\@gls@xtr@current@style</code>	35, 228
87, 92	<code>\@gls@xtr@currentunitcount</code>	77–80
<code>\@gls@org@glsnoidx@displayloc</code>	<code>\@gls@xtr@currunitcount</code>	79, 81
91	<code>\@gls@xtr@declareoption</code>	4, 11, 14
<code>\@gls@org@glsseeformat</code>	<code>\@gls@xtr@defaultnoglossarywarning</code> ..	14
91	<code>\@gls@xtr@deprecated@abbrstyle</code>	
<code>\@gls@preglossaryhook</code> 173, 174, 176, 177, 182–184	
89, 132	<code>\@gls@xtr@disabledflycommand</code>	34
<code>\@gls@prevlevel</code>	<code>\@gls@xtr@display@loc</code>	93
227	<code>\@gls@xtr@do@wrindex</code>	58
<code>\@gls@quotechar</code>	<code>\@gls@xtr@do@glsdisablehyperinlist</code> ..	56
127	<code>\@gls@xtr@do@redef@forglsentries</code>	6
<code>\@gls@reference</code>	<code>\@gls@xtr@do@style</code>	15, 210
30, 85, 86	<code>\@gls@xtr@do@titlecaps@warn</code> .	121–124, 132
<code>\@gls@saveentrycounter</code>	<code>\@gls@xtr@doabbreviationsdef</code>	11
9, 16, 41	<code>\@gls@xtr@doaccsupp</code>	14, 15
<code>\@gls@see@noindex</code>	<code>\@gls@xtr@docdefval</code>	9, 10, 31
98, 99	<code>\@gls@xtr@docounterrecord</code>	7
<code>\@gls@setdefault@glslink@opts</code>	<code>\@gls@xtr@doloctag</code>	37
57	<code>\@gls@xtr@dorecord</code>	7
<code>\@gls@short</code>	<code>\@gls@xtr@dostylewarn</code>	155
139	<code>\@gls@xtr@enabletagging</code>	131
<code>\@gls@shortpl</code>	<code>\@gls@xtr@end@</code>	31
138–140	<code>\@gls@xtr@endescpsh</code>	128–130
<code>\@gls@sort</code>		
103		
<code>\@gls@tmpb</code>		
130		
<code>\@gls@type</code>		
86–88, 155, 219–225		
<code>\@gls@write@entrycounts</code>		
72		
<code>\@gls@write@entryunitcounts</code>		
80		
<code>\@gls@write@entryunitcounts@do</code>		
81		
<code>\@gls@abbrv@current@abbreviation</code> 139, 152		
<code>\@gls@acronymlists</code>		
82		
<code>\@gls@disp</code>		
41		
<code>\@gls@doifexistsorwarn</code> ...		
10, 122, 123, 125		
<code>\@gls@entry</code>		
72, 81		
<code>\@gls@link</code>		
60		
<code>\@gls@locref</code>		
7		
<code>\@gls@nextpages</code>		
89		
<code>\@gls@nonextpages</code>		
89		
<code>\@gls@numberformat</code>		
6, 7, 41, 57, 126, 127		
<code>\@gls@order</code>		
85		
<code>\@gls@pl@</code>		
61, 73, 75		
<code>\@gls@plural@</code>		
61		
<code>\@gls@punc@token</code>		
136		
<code>\@gls@style@alttree</code>		
217		
<code>\@gls@style@inline</code>		
217		
<code>\@gls@style@listdotted</code>		
212		

<code>\@glsxtr@entrycount@org@localreset</code>	<code>\@glsxtr@org@Glsxtrtitlefirstplural</code>
..... 71, 72 193, 195
<code>\@glsxtr@entrycount@org@localunset</code> .. 71	<code>\@glsxtr@org@Glsxtrtitlefull</code> .. 194, 195
<code>\@glsxtr@entrycount@org@reset</code> 71	<code>\@glsxtr@org@Glsxtrtitlefullpl</code> 194, 195
<code>\@glsxtr@entrycount@org@unset</code> 71	<code>\@glsxtr@org@Glsxtrtitlelong</code> .. 194, 195
<code>\@glsxtr@entryunitcount@org@localreset</code>	<code>\@glsxtr@org@Glsxtrtitlelongpl</code> 194, 195
..... 80	<code>\@glsxtr@org@Glsxtrtitleplural</code> 193, 194
<code>\@glsxtr@entryunitcount@org@localunset</code>	<code>\@glsxtr@org@Glsxtrtitleshort</code> .. 193, 194
..... 79	<code>\@glsxtr@org@Glsxtrtitleshortpl</code> 193, 194
<code>\@glsxtr@entryunitcount@org@reset</code> .. 79	<code>\@glsxtr@org@Glsxtrtitletext</code> .. 193, 194
<code>\@glsxtr@entryunitcount@org@unset</code> .. 79	<code>\@glsxtr@org@MakeUppercase</code> 193, 194
<code>\@glsxtr@err@undefaction</code> 6, 9	<code>\@glsxtr@org@checkfirsthyper</code> 55, 84
<code>\@glsxtr@field@linkdefs</code> 39	<code>\@glsxtr@org@delimN</code> 37, 38
<code>\@glsxtr@format@overridefalse</code> 126	<code>\@glsxtr@org@delimR</code> 37, 38
<code>\@glsxtr@format@overridetrue</code> .. 126, 127	<code>\@glsxtr@org@doseeglossary</code> 86
<code>\@glsxtr@foundinlist</code> 137	<code>\@glsxtr@org@gls@</code> 40
<code>\@glsxtr@full</code> 143	<code>\@glsxtr@org@glsdisp</code> 41
<code>\@glsxtr@fullpl</code> 144	<code>\@glsxtr@org@glsignore</code> 37, 38
<code>\@glsxtr@gettype</code> 88	<code>\@glsxtr@org@glspl@</code> 40
<code>\@glsxtr@glossdescfont</code> 121, 122	<code>\@glsxtr@org@glsxtrtitlefirst</code> .. 193, 194
<code>\@glsxtr@glossnamefont</code> 122–126	<code>\@glsxtr@org@glsxtrtitlefirstplural</code>
<code>\@glsxtr@gobbleto@endescspch</code> 130 193, 195
<code>\@glsxtr@idx@displaynumberlist</code> 87	<code>\@glsxtr@org@glsxtrtitlefull</code> .. 194, 195
<code>\@glsxtr@idx@entrynumberlist</code> 87	<code>\@glsxtr@org@glsxtrtitlefullpl</code> 194, 195
<code>\@glsxtr@ifcsstart</code> 31	<code>\@glsxtr@org@glsxtrtitlelong</code> .. 193, 195
<code>\@glsxtr@ifpunctoken</code> 137	<code>\@glsxtr@org@glsxtrtitlelongpl</code> 193, 195
<code>\@glsxtr@ifunitcounter</code> 76	<code>\@glsxtr@org@glsxtrtitleplural</code> 193, 194
<code>\@glsxtr@insert@dots</code> 138	<code>\@glsxtr@org@glsxtrtitleshort</code> .. 193, 194
<code>\@glsxtr@insert@dots@next</code> 138, 139	<code>\@glsxtr@org@glsxtrtitleshortpl</code> 193, 194
<code>\@glsxtr@insert@dots</code> 139	<code>\@glsxtr@org@glsxtrtitletext</code> .. 193, 194
<code>\@glsxtr@label</code> 30, 120	<code>\@glsxtr@org@makeglossaries</code> 85
<code>\@glsxtr@loadstyles</code> 211	<code>\@glsxtr@org@markboth</code> 192, 193
<code>\@glsxtr@longnewglossaryentry</code> 23	<code>\@glsxtr@org@markright</code> 192, 193
<code>\@glsxtr@mixed@assign@sortkey</code> 88	<code>\@glsxtr@org@newacronymstyle</code> 84
<code>\@glsxtr@noidx@displaynumberlist</code> .. 87	<code>\@glsxtr@org@postdescription</code> 133
<code>\@glsxtr@noidx@do</code> 103	<code>\@glsxtr@org@see@noindex</code> 98, 99
<code>\@glsxtr@noidx@entrynumberlist</code> 87	<code>\@glsxtr@org@setacronymstyle</code> 84
<code>\@glsxtr@noidx@getgrouptitle</code> 101	<code>\@glsxtr@org@printglossary</code> 34, 90
<code>\@glsxtr@noidx@numberlistloop</code> 87	<code>\@glsxtr@org@warndep</code> 138
<code>\@glsxtr@noop@recordcounter</code> 7, 9	<code>\@glsxtr@p@acrlong@</code> 61
<code>\@glsxtr@notfoundinlist</code> 137	<code>\@glsxtr@p@acrlongpl@</code> 61
<code>\@glsxtr@op@recordcounter</code> 9	<code>\@glsxtr@p@acrshort@</code> 61
<code>\@glsxtr@optlist</code> 34	<code>\@glsxtr@p@acrshortpl@</code> 61
<code>\@glsxtr@org@GLS@</code> 40	<code>\@glsxtr@p@long@</code> 61
<code>\@glsxtr@org@GLSpl@</code> 40	<code>\@glsxtr@p@longpl@</code> 61
<code>\@glsxtr@org@Gls@</code> 40	<code>\@glsxtr@p@plural@</code> 61
<code>\@glsxtr@org@Glspl@</code> 40	<code>\@glsxtr@p@short@</code> 61
<code>\@glsxtr@org@Glsxtrtitlefirst</code> .. 193, 194	<code>\@glsxtr@p@shortpl@</code> 61
	<code>\@glsxtr@p@text@</code> 61

<code>\@glstr@pagetag</code>	37	<code>\@ifpackageloaded</code>	
<code>\@glstr@pagetag</code>	37	...	4, 11, 100, 104, 120, 122, 125, 126, 209
<code>\@glstr@prevunitcount</code>	79	<code>\@ifstar</code>	17, 23–25, 31, 58, 101, 131
<code>\@glstr@printglossopts</code>	34, 88, 90	<code>\@ifundefined</code>	209
<code>\@glstr@provide@addstoragekey</code>	18	<code>\@ignored@glossaries</code>	24–26
<code>\@glstr@provide@storagekey</code>	17	<code>\@input</code>	98
<code>\@glstr@record</code>	9, 39–41	<code>\@input@</code>	94
<code>\@glstr@redef@for@glstr@entries</code>	6, 15, 16	<code>\@istfilename</code>	85
<code>\@glstr@redef@styles</code>	14, 15, 210	<code>\@makeglossary</code>	85, 86
<code>\@glstr@reg@glosslist</code>	85–88, 90	<code>\@mfu@domakefirstuc</code>	132
<code>\@glstr@s@longnewglossaryentry</code>	23	<code>\@mfu@nocaplist</code>	132
<code>\@glstr@save@preloctag</code>	37, 38	<code>\@one</code>	72, 81
<code>\@glstr@set@entrycount@unset@attr</code>	70	<code>\@newglossaryentry@def@counters</code>	71, 78
<code>\@glstr@set@entryunitcount@unset@attr</code>	82	<code>\@newglossaryentry@posthook</code>	8, 17, 53
<code>\@glstr@setup@shortcuts</code>	13, 15, 16	<code>\@newglossaryentry@prehook</code>	8, 17, 23, 53
<code>\@glstr@shortcuts@val</code>	13, 100	<code>\@nil</code>	103
<code>\@glstr@swaptwo</code>	137	<code>\@nnil</code>	127, 128, 130, 137, 138
<code>\@glstr@tag</code>	132	<code>\@no@glstr@index@aliased</code>	57
<code>\@glstr@tagging@cs</code>	132	<code>\@no@makeglossaries</code>	98
<code>\@glstr@this@loctag</code>	37, 38	<code>\@nopostdesc</code>	89
<code>\@glstr@tmp</code>	14, 15	<code>\@onelevel@sanitize</code>	34
<code>\@glstr@type</code>	120	<code>\@onlypreamble</code>	34, 37, 81, 99, 100, 127, 129, 131
<code>\@glstr@unitcountlist</code>	77	<code>\@org@glossaryentry@numbers</code>	89
<code>\@glstr@unsrt@get@group@title</code>	101	<code>\@org@newglossaryentry@prehook</code>	23
<code>\@glstr@use@see</code>	29	<code>\@print@unsrt@glossary</code>	101
<code>\@glstr@warn@on@exists@ordo</code>	6, 9	<code>\@printgloss@set@sort</code>	88, 89
<code>\@glstr@warn@undefaction</code>	6, 9	<code>\@printglossary</code>	34, 101
<code>\@glstr@doc@deffalse</code>	30	<code>\@printunsrt@glossary</code>	101
<code>\@glstr@entry@fmt</code>	18	<code>\@sGlsXtrEnableOnTheFly</code>	31
<code>\@glstr@index@aliased</code>	56	<code>\@secondofthree</code>	42–44, 49, 51–54, 144, 145, 147, 148, 150, 151
<code>\@glstr@index@cross@refs@false</code>	10	<code>\@secondoftwo</code>	41, 45–53, 55, 60, 84, 90, 137, 143, 144, 146–152, 164, 193, 194
<code>\@glstr@index@cross@ref@true</code>	10	<code>\@sglstr@provide@storagekey</code>	17
<code>\@glstr@in@mark</code>	192, 193	<code>\@thirdofthree</code>	42–44, 50–53, 55, 144, 145, 147, 149, 150, 152
<code>\@glstr@long</code>	61, 147	<code>\@thirdoftwo</code>	45–49
<code>\@glstr@long@pl</code>	61, 151	<code>\@warn@nomakeglossaries</code>	95
<code>\@glstr@not@in@mark</code>	192, 193	<code>\@xdy@main@language</code>	94
<code>\@glstr@rp</code>	66, 67	<code>\@xdy@language</code>	94
<code>\@glstr@rp@opt</code>	64		
<code>\@glstr@rpl</code>	33, 34		
<code>\@glstr@rpost@loctag</code>	36–38		
<code>\@glstr@rpre@loctag</code>	36–38		
<code>\@glstr@rset@alias@noindex</code>	56, 57		
<code>\@glstr@rshort</code>	61, 146		
<code>\@glstr@rshort@pl</code>	61, 149		
<code>\@glstr@rundef@tag</code>	5, 16		
<code>\@gobble</code>	6, 9, 11, 42, 138		
<code>\@gobbletwo</code>	138		
<code>\@ifnextchar</code>	58		
		<code>_</code>	96, 97
		A	
		<code>\AB</code>	12
		<code>\Ab</code>	12
		<code>\ab</code>	11
		abbreviation styles:	
		long-noshort	182

long-postshort-user	187	\AtEndDocument	29, 72, 80, 94
long-short-user	186		
short	167	B	
short-long-user	188	babel package	127, 128, 136
short-postlong-user	190	\begin	92, 96, 101, 212–216
\abbreviationsname	11	\begingroup	6, 57, 101
\abbrvpluralsuffix		\bgroup	23, 89
.... 100, 139, 140, 158, 160, 162, 164,			
166, 167, 169, 171–177, 185, 186, 189, 191		C	
\ABP	12	\catcode	98
\Abp	12	category attributes:	
\abp	11	discardperiod	135
\ACRfullfmt	83	entrycount	69–72, 81, 82
\Acrfullfmt	83	firsttuc	124
\acrfullfmt	83	glossdesc	120
\ACRfullplfmt	83	glossdescfont	121
\Acrfullplfmt	83	glossname	122
\acrfullplfmt	83	glossnamefont	122, 125
\acronymentry	83	headuc	195
\acronymfont	49–53, 63, 84	indexname	127
\acronymname	11	indexonlyfirst	57
\acronymsort	83	insertdots	139, 140
\acronymtype	11, 83, 84	nohyper	56
\acrpluralsuffix	83, 100	nohyperfirst	42–44
\actualchar	130	regular	39, 76, 157–
\addtolength	227	162, 164, 166, 168, 169, 179, 180, 185, 191	
\advance	72, 81, 99	\cGLS	12, 70, 82
\AF	12	\cGls	12, 70, 82
\Af	12	\cglS	11, 70, 82
\af	12	\cGLSformat	74
\AFP	12	\cGlsformat	74
\Afp	12	\cglSformat	73, 75
\afp	12	\cGLSpl	12, 70, 82
\AL	12	\cGlspl	12, 70, 82
\Al	12	\cglSpl	11, 70, 82
\al	11	\cGLSplformat	74
\ALP	12	\cGlsplformat	74
\Alp	12	\cglSplformat	73, 75
\alp	12	\columnwidth	35, 36
\AnyTrackedLanguages	209	\count@	72, 81
\appto	8, 15,	\csappto	21
17, 22, 28, 53, 58, 71, 78, 102, 126, 136, 139		\csdef	17, 20–22, 53–55, 72, 77,
\AS	12	78, 80, 115, 155–157, 164, 186, 188, 190, 212	
\As	12	\cseappto	26
\as	11	\csedef	20, 78
\ASP	12	\csgdef	21, 24–26, 30, 37, 71, 72, 77, 80
\Asp	12	\cslet	20, 23, 89
\asp	11	\csletcs	20, 21, 156, 157
\AtBeginDocument	16, 35, 36, 100		

<code>\csname</code>	5, 6, 18, 25, 35, 38, 41, 49–55, 57, 64, 77, 78, 86, 94, 97, 98, 101, 103, 120, 138, 143–152, 157, 226	<code>\egroup</code>	23, 89
<code>\cspreto</code>	22	<code>\else</code>	6, 7, 10–14, 21, 30, 32, 36, 38, 57, 58, 73, 85, 88–90, 93, 96, 97, 99, 126–128, 130, 137–139, 146–152, 158– 169, 185–187, 189, 191, 213–217, 227, 228
<code>\csuse</code>	18, 19, 22, 25, 37, 54, 55, 66–68, 76–80, 89, 92, 93, 101–103, 115, 126, 134, 155, 157, 219–222	<code>\emph</code>	178
<code>\csxdef</code>	28, 77, 80	<code>\empty</code>	93
<code>\currentglossary</code>	89	<code>\encapchar</code>	130
<code>\CurrentOption</code>	15, 211	<code>\end</code>	92, 96, 97, 102, 212–216
<code>\CurrentTrackedLanguageTag</code>	99, 100	<code>\end@glxtr@display@loc</code>	93
<code>\CurrentTrackedTag</code>	210	<code>\endcsname</code>	5, 6, 18, 25, 35, 38, 41, 49–55, 57, 64, 77, 78, 86, 94, 97, 98, 101, 103, 120, 138, 143–152, 157, 226
<code>\CustomAbbreviationFields</code>	140, 158–162, 164, 165, 167, 168, 170, 178, 180, 182, 185–188, 190	<code>\endgroup</code>	7, 57, 101
D		entry categories:	
<code>\DeclareAcronymList</code>	83	abbreviation	152
<code>\DeclareOption</code>	4, 211	general	115, 117
<code>\DeclareOptionX</code>	4, 15	index	118
<code>\def</code> . 6, 8, 9, 16, 23, 25, 29–34, 36, 38–53, 57, 59–64, 73–75, 83, 86, 88–93, 101, 102, 126–130, 132, 136–140, 143–152, 155, 227		<code>\epreto</code>	127
<code>\defglstryfmt</code>	24–26	<code>\equal</code>	97, 98
<code>\define@boolkey</code>	10, 56	etoolbox package	4
<code>\define@choicekey</code>	6, 8, 10, 13, 14, 90	<code>\expandafter</code>	15, 18, 29, 31–33, 54, 55, 58, 59, 64, 75, 76, 86–88, 90, 93, 101, 103, 120, 123, 124, 127, 130, 137, 139, 140
<code>\define@key</code>	8, 14, 15, 17, 53, 138	<code>\expandonce</code>	83, 127, 128
<code>\DefineAcronymSynonyms</code>	13	F	
<code>\delimN</code>	37, 38	<code>\fi</code>	6, 7, 9–14, 21, 29–32, 35, 36, 38, 57, 58, 73, 80, 81, 85, 88–90, 93, 95–99, 126–128, 130, 137, 139, 146–152, 158– 169, 185–187, 189, 191, 213–217, 219–228
<code>\delimR</code>	37, 38	first use	229
<code>\detokenize</code>	31	flag	229
<code>\dimen@</code>	85, 219–226	text	229
<code>\dimen@i</code>	220–222	<code>\firstacronymfont</code>	84, 85
<code>\dimen@ii</code>	220–222	fontspec package	100
<code>\dimexpr</code>	35, 36, 218	<code>\footnote</code>	162
<code>\disable@keys</code>	11, 16, 30	<code>\forall glossaries</code>	29, 101, 118, 120, 219–225
<code>\do</code>	5, 14, 30, 70, 82, 85, 88, 101, 120, 131	<code>\forall glsentries</code>	72, 81
<code>\do@gl@link@checkfirsthyper</code>	18, 39, 41, 49–53, 143–152	<code>\ForEachTrackedDialect</code>	209
<code>\do@gl@disablehyperinlist</code>	56	<code>\for glsentries</code>	5, 29, 118, 120, 219–225
doc package	130	<code>\for listcsloop</code>	19, 81, 92
<code>\dolistcsloop</code>	19	<code>\for listloop</code>	91, 132
<code>\DTLifinlist</code>	86, 87, 90	<code>\futurelet</code>	136
E		G	
<code>\eappto</code>	7, 14, 24–26, 127, 211	<code>\gdef</code>	37, 129
<code>\edef</code>	5, 6, 22, 24–27, 41, 55, 57, 76–80, 85–87, 90, 93, 94, 98, 121, 122, 124, 125, 127, 128, 130, 138, 220–222	<code>\Genacrfullformat</code>	83
<code>\eglssetwidest</code>	219–226	<code>\genacrfullformat</code>	83
		<code>\GenericAcronymFields</code>	83

<code>\Genplacrfullformat</code>	83	<code>\gls@defglossaryentry</code>	23, 32, 33
<code>\genplacrfullformat</code>	83	<code>\gls@dotoctitle</code>	89
<code>\glo@grabfirst</code>	103	<code>\gls@level</code>	103
<code>\glo@name</code>	123, 124	<code>\gls@noidxglossary</code>	86
<code>\gloaliaslabel</code>	60	<code>\gls@org@glossaryentryfield</code>	89
<code>\global</code>	23, 89, 103	<code>\gls@org@glossarysubentryfield</code>	89
<code>\glolinkprefix</code>	60, 100, 102	<code>\gls@save@numberlist</code>	36, 38
glossaries package	211	<code>\gls@tmplen</code>	219–225, 227
glossaries-accsupp package	14, 15, 104	<code>\gls@type</code>	86
glossaries-extra package	2	<code>\glsabbrvdefaultfont</code>	142, 158, 160, 162, 164, 166, 167, 169
glossaries-extra-stylemods package	14, 133, 210	<code>\glsabbrvmfont</code>	178–184
glossaries.sty package	23	<code>\glsabbrvfont</code>	62, 63, 84, 142, 146, 147, 149, 150, 153, 158–162, 164–167, 169–191
<code>\GlossariesExtraWarning</code>	5, 11, 21, 22, 32, 34, 84, 87, 93, 96, 98, 101, 121, 122, 124, 125, 132, 157	<code>\glsabbrvuserfont</code>	184–186, 189, 191
<code>\GlossariesExtraWarningNoLine</code>	11, 72, 81	<code>\glsabrvfont</code>	158, 160, 162, 164, 179, 180, 185, 186, 188, 190, 191
<code>\GlossariesWarning</code>	37, 87, 89, 91, 92, 155	<code>\GLSaccessdesc</code>	45
<code>\GlossariesWarningNoLine</code>	86, 95	<code>\Glsaccessdesc</code>	45, 121, 131
glossary styles:		<code>\glsaccessdesc</code>	45, 121, 135
alttree	217, 218, 226	<code>\GLSaccessdescplural</code>	46
inline	217	<code>\Glsaccessdescplural</code>	46
listdotted	212	<code>\glsaccessdescplural</code>	46
listdottedstyle	212	<code>\GLSaccessfirst</code>	43
sublistdotted	212	<code>\Glsaccessfirst</code>	43
glossary-long package	213	<code>\glsaccessfirst</code>	43
glossary-longbooktabs package	213	<code>\GLSaccessfirstplural</code>	44
glossary-mcols package	211	<code>\Glsaccessfirstplural</code>	44
glossary-tree package	211	<code>\glsaccessfirstplural</code>	44
<code>\glossaryentrynumbers</code>	38, 89, 103, 104	<code>\Glsaccesslong</code>	52, 141, 148, 159, 166, 169, 186, 187
<code>\glossaryheader</code>	92, 101, 212–216, 227	<code>\glsaccesslong</code>	51, 52, 141, 148, 149, 158, 160, 162, 163, 165–169, 185, 187, 189, 191
<code>\glossaryname</code>	89	<code>\Glsaccesslongpl</code>	53, 141, 151, 159, 166, 169, 186, 187
<code>\glossarypostamble</code>	92, 102	<code>\glsaccesslongpl</code>	52, 53, 141, 151, 152, 158, 160, 161, 163, 165–169, 185, 187, 189, 191
<code>\glossarypreamble</code>	92, 101	<code>\GLSaccessname</code>	45
<code>\glossarysection</code>	92, 97, 101, 102	<code>\Glsaccessname</code>	45
<code>\glossarytitle</code>	25, 88, 89, 92, 97, 101	<code>\glsaccessname</code>	45
<code>\glossarytoctitle</code>	25, 89, 92, 97, 101	<code>\GLSaccessplural</code>	44
<code>\glossentry</code>	89, 103, 104, 212–216, 227	<code>\Glsaccessplural</code>	43
<code>\glossentrydesc</code>	212–218	<code>\glsaccessplural</code>	43
<code>\glossentryname</code>	212–216, 227	<code>\Glsaccessshort</code>	50, 147, 153, 160, 163, 165, 168, 189, 191
<code>\glossentrysymbol</code>	213–218	<code>\glsaccessshort</code>	49, 50, 141, 146, 147, 153, 158–160, 162–169, 185–187, 189, 191
<code>\glossxtrsetpopts</code>	133	<code>\Glsaccessshortpl</code>	51, 150, 153, 161, 163, 165, 168, 189, 191
<code>\GLS</code>	70, 82		
<code>\Gls</code>	33, 70, 82		
<code>\gls</code>	21, 32, 34, 70, 82, 87, 95		
<code>\gls@assign@desc</code>	23		
<code>\gls@assign@field</code>	8, 17, 53		
<code>\gls@checkseeallowed</code>	31, 86		
<code>\gls@codepage</code>	94		
<code>\gls@defdocnewglossaryentry</code>	71, 78		

<code>\glsaccesssshortpl</code>	50, 51, 141, 149, 150, 153, 158–160, 163–169, 185–187, 189, 191
<code>\GLSAccessssymbol</code>	46
<code>\Glsaccessssymbol</code>	46, 131
<code>\glsaccessssymbol</code>	46, 131, 135
<code>\GLSAccessssymbolplural</code>	47
<code>\Glsaccessssymbolplural</code>	47
<code>\glsaccessssymbolplural</code>	47
<code>\GLSAccessstext</code>	42
<code>\Glsaccessstext</code>	42
<code>\glsaccessstext</code>	42
<code>\glsacrshortcutsrue</code>	13
<code>\glsacspacemax</code>	85
<code>\glsadd</code>	30, 95
<code>\glsaddstoragekey</code>	22, 115
<code>\glsbackslash</code>	32
<code>\glscapscase</code>	41–55, 143–154
<code>\glscategory</code>	39, 42, 55, 62, 63, 116, 117, 120–122, 124–126, 131, 134, 143–147, 149, 150
<code>\glscategorylabel</code>	55, 138–140, 164, 186, 188, 190
<code>\glsclosebrace</code>	96–98
<code>\glscurrententrylabel</code>	36–38, 89, 101, 102, 126, 133, 134
<code>\glscurrentfieldvalue</code>	18, 19, 184
<code>\glscustomtext</code>	39, 49–53, 143–152, 154
<code>\glsdefaulttype</code>	5, 11, 21, 22, 88, 95, 101, 102
<code>\glsdescriptionaccessdisplay</code>	108, 109, 121
<code>\glsdescriptionpluralaccessdisplay</code>	109
<code>\glsdescwidth</code>	212, 214–216
<code>\glsdetoklabel</code>	19–21, 23, 27–31, 41, 57, 60, 71, 72, 77–81, 86, 89–91, 103, 120, 123, 124, 220–222
<code>\glsdisplaynumberlist</code>	87, 90
<code>\glsdohyperlink</code>	60
<code>\glsdohypertarget</code>	90
<code>\glsdoifexists</code>	10, 20, 26, 28, 39, 41, 49–53, 91, 92, 143–152
<code>\glsdoifexistsordo</code>	18, 19, 41
<code>\glsdoifexistsorwarn</code>	10, 120, 121, 130, 131
<code>\glsdoifnoexists</code>	23
<code>\glsdonohyperlink</code>	60
<code>\glsdosanitizesort</code>	87
<code>\glsenableentrycount</code>	70, 72, 80
<code>\glsenableentryunitcount</code>	72, 81, 82
<code>\glsentrycurrcount</code>	71, 72, 79
<code>\Glsentrydesc</code>	108, 113, 122
<code>\glsentrydesc</code>	108, 109, 113, 122
<code>\Glsentrydescplural</code>	109, 113
<code>\glsentrydescplural</code>	109, 113, 114
<code>\Glsentryfirst</code>	76, 106, 112
<code>\glsentryfirst</code>	76, 106, 112, 206
<code>\Glsentryfirstplural</code>	76, 107, 112
<code>\glsentryfirstplural</code>	76, 107, 112, 113, 206
<code>\glsentryfmt</code>	24–26
<code>\Glsentryfull</code>	83
<code>\glsentryfull</code>	83
<code>\Glsentryfullpl</code>	83
<code>\glsentryfullpl</code>	83
<code>\glsentryitem</code>	212–216, 227
<code>\Glsentrylong</code>	63, 64, 76, 111, 114
<code>\glsentrylong</code>	63, 64, 76, 111, 114, 164, 188, 190, 207
<code>\Glsentrylongpl</code>	63, 64, 76, 111, 114
<code>\glsentrylongpl</code>	63, 64, 76, 111, 114, 207, 208
<code>\Glsentryname</code>	104, 112, 123–126
<code>\glsentryname</code>	104, 105, 111, 112, 127, 219–226
<code>\glsentrynumberlist</code>	87, 92, 224–226
<code>\Glsentryplural</code>	106, 112
<code>\glsentryplural</code>	105, 106, 112, 205
<code>\glsentryprevcount</code>	71, 73, 79
<code>\glsentryprevmaxcount</code>	79
<code>\glsentryprevtotalcount</code>	79
<code>\Glsentryshort</code>	62, 63, 110, 114
<code>\glsentryshort</code>	62, 63, 85, 109, 110, 114, 186, 188, 203, 204
<code>\Glsentryshortpl</code>	62, 63, 110, 114
<code>\glsentryshortpl</code>	62, 63, 110, 114, 204
<code>\Glsentrysymbol</code>	107, 113
<code>\glsentrysymbol</code>	107, 113, 223, 224
<code>\Glsentrysymbolplural</code>	108, 113
<code>\glsentrysymbolplural</code>	108, 113
<code>\Glsentrytext</code>	105, 112
<code>\glsentrytext</code>	105, 112, 204, 205
<code>\Glsentryuseri</code>	47
<code>\glsentryuseri</code>	47
<code>\Glsentryuserii</code>	47
<code>\glsentryuserii</code>	48
<code>\Glsentryuseriii</code>	48
<code>\glsentryuseriii</code>	48
<code>\Glsentryuseriv</code>	48
<code>\glsentryuseriv</code>	48
<code>\Glsentryuserv</code>	48
<code>\glsentryuserv</code>	49
<code>\Glsentryuservi</code>	49
<code>\glsentryuservi</code>	49
<code>\glsfieldfetch</code>	60

<code>\glsfieldxdef</code>	120	<code>\glsinlinedescformat</code>	217
<code>\glsfindwidesttoplevelname</code>	219	<code>\glsinlinesubdescformat</code>	217
<code>\GLSfirst</code>	198, 199	<code>\glsinsert</code>	41, 49–53, 143–154
<code>\Glsfirst</code>	199	<code>\glskeylisttok</code>	83, 139, 140
<code>\glsfirst</code>	198	<code>\glslabel</code>	27, 39, 55–57, 59, 60, 84, 134, 135, 152–154, 164, 186, 188, 190
<code>\glsfirstabbrvdefaultfont</code>		<code>\glslabeltok</code>	
.....	142, 158, 160, 162, 164, 166, 167, 169	83, 139, 140, 158–162, 164, 166– 168, 170, 178–180, 182, 185–188, 190, 191
<code>\glsfirstabbrvfont</code>	179–184	<code>\glsletentryfield</code>	127
<code>\glsfirstabbrvfont</code>	84, 141, 158–169, 171–177, 179–187, 189, 191	<code>\glslink</code>	83
<code>\glsfirstabbrvuserfont</code> ..	185, 186, 188–191	<code>\glslink options</code>	
<code>\glsfirstaccessdisplay</code>	106	format	126
<code>\glsfirstlongdefaultfont</code>		hyper	192
.....	158, 160, 166, 167, 169	noindex	6, 56, 192
<code>\glsfirstlongfont</code>	179–183	<code>\glslinkcheckfirsthyperhook</code>	55
<code>\glsfirstlongfont</code>	141, 158–162, 164, 166–170, 178–183, 185–191	<code>\glslinkvar</code>	58, 59
<code>\glsfirstlongfootnotefont</code>	162–165	<code>\glslistdottedwidth</code>	212
<code>\glsfirstlonguserfont</code> ..	185, 186, 189, 191	<code>\glslongaccessdisplay</code>	111
<code>\GLSfirstplural</code>	199	<code>\glslongdefaultfont</code>	
<code>\Glsfirstplural</code>	199, 200	142, 158, 160, 161, 166, 167, 169
<code>\glsfirstplural</code>	199	<code>\glslongfont</code>	178–183
<code>\glsfirstpluralaccessdisplay</code> ..	106, 107	<code>\glslongfont</code>	63, 142, 148, 149, 151, 152, 158, 160, 162, 164, 166, 167, 169, 179–183, 185, 186, 189, 191
<code>\glsforeachincategory</code>	155	<code>\glslongfootnotefont</code>	161, 162, 164
<code>\glsngenentryfmt</code>	39	<code>\glslongpltok</code>	140, 158–162, 168, 170, 179, 180, 182, 185–188, 190, 191
<code>\glsgetattribute</code>		<code>\glslongpluralaccessdisplay</code>	111
.....	59, 60, 73, 77–79, 121, 122, 124, 125, 127	<code>\glslongtok</code> ..	83, 139, 140, 158–162, 164, 166– 168, 170, 178–180, 182, 185–188, 190, 191
<code>\glsgetcategoryattribute</code>	116	<code>\glslonguserfont</code> ...	185, 186, 188, 189, 191
<code>\glsgetwidestname</code>	218	<code>\glsnameaccessdisplay</code> ..	104, 105, 123, 125
<code>\glsgroupheading</code>	103, 212–216, 227	<code>\glsnamefont</code>	122–126
<code>\glsgroupskip</code>	103, 213–217, 228	<code>\glsnextpages</code>	89
<code>\glshasattribute</code>	59, 72, 73, 77–81, 121, 122, 124, 125, 127, 158– 162, 164, 179, 180, 185, 186, 188, 190, 191	<code>\glsnoidxdisplayloc</code>	91
<code>\glshascategoryattribute</code>	116	<code>\glsnoidxdisplaylocclsthandler</code>	91
<code>\glshypernumber</code>	126	<code>\glsnoidxlolist</code>	92, 103, 104
<code>\glsifattribute</code>	42, 56, 58, 66, 118, 121–124, 133, 135, 136, 195–203	<code>\glsnoidxnumberlistloophandler</code>	91
<code>\glsifcategory</code>	118	<code>\glsnonextpages</code>	89
<code>\glsifcategoryattribute</code>		<code>\glsnonumberlistfalse</code>	36
.....	55, 116, 117, 139, 140	<code>\glsnonumberlisttrue</code>	36
<code>\glsifnotregular</code>	42	<code>\glsnumberlistloop</code>	87
<code>\glsifnotregularcategory</code>	117	<code>\glsnumlistlastsep</code>	91
<code>\glsifplural</code>	41, 43, 44, 46, 47, 49–53, 135, 136, 143–153	<code>\glsnumlistsep</code>	90
<code>\glsifregular</code>	39, 42, 76	<code>\glsopenbrace</code>	96–98
<code>\glsifregularcategory</code>	117	<code>\glsorder</code>	85
<code>\glsifusetranslator</code>	25	<code>\glspagelistwidth</code>	212, 214–216
<code>\glsignore</code>	37, 38	<code>\glspar</code>	102

<code>\GLSpl</code>	70, 82	<code>\glxtr</code>	34
<code>\Glspl</code>	33, 70, 82	<code>\glxtr@@do@wrglossary</code>	7, 9
<code>\glspl</code>	33, 70, 82	<code>\glxtr@addloclistfield</code>	9
<code>\GLSplural</code>	197, 198	<code>\glxtr@addunused</code>	29
<code>\Glsplural</code>	198	<code>\glxtr@applyabbrvfmt</code>	152
<code>\glsplural</code>	197, 198	<code>\glxtr@applyabbrvstyle</code>	138, 139, 155
<code>\glspluralaccessdisplay</code>	105, 106	<code>\glxtr@counterrecord</code>	101
<code>\glspluralsuffix</code>	100, 139, 142	<code>\glxtr@doption</code>	4, 11, 15
<code>\glspostdescription</code>	133, 212–218	<code>\glxtr@fields</code>	99
<code>\glspostinline</code>	217	<code>\glxtr@headentry@p</code>	66, 67
<code>\glspostlinkhook</code>	40, 41, 49–53, 64, 143–152	<code>\glxtr@ifnextpunc</code>	136
<code>\glsprestandardsort</code>	87	<code>\glxtr@ifpunctoken</code>	136
<code>\glsresetentrylist</code>	92, 101	<code>\glxtr@indexonly@saveentrycounter</code>	9, 16
<code>\glssee</code>	22	<code>\glxtr@keylist</code>	32, 33
<code>\glsseeformat</code>	29, 86, 91	<code>\glxtr@langtag</code>	100
<code>\glssetabbrvfmt</code>	39, 42, 62, 63, 120–122, 124, 125, 131, 143–147, 149, 150	<code>\glxtr@linkprefix</code>	100
<code>\glssetattribute</code>	158–162, 164, 166–168, 170, 179, 180, 182, 185, 186, 188, 190, 191	<code>\glxtr@makeglossaries</code>	85
<code>\glssetcategoryattribute</code>	71, 82, 84, 116, 117, 119, 131	<code>\glxtr@newabbreviation</code>	84, 139
<code>\glssetnoexpandfield</code>	8	<code>\glxtr@next</code>	137
<code>\glssettoctitle</code>	89	<code>\glxtr@org@newignoredglossary</code>	24
<code>\glsshortaccessdisplay</code>	109, 110	<code>\glxtr@orgmakenoidxglossaries</code>	30
<code>\glsshortpltok</code>	140, 158–162, 164–167, 179, 180, 185, 186, 188, 190, 191	<code>\glxtr@pluralsuffixes</code>	100
<code>\glsshortpluralaccessdisplay</code>	110	<code>\glxtr@provideignoredglossary</code>	25
<code>\glsshorttok</code>	83, 139, 140, 158–162, 164, 165, 167, 170, 178–180, 182, 185–188, 190, 191	<code>\glxtr@punctlist</code>	136, 137
<code>\glssubentryitem</code>	212–217, 227	<code>\glxtr@record</code>	7
<code>\glssymbolaccessdisplay</code>	107	<code>\glxtr@resource</code>	98
<code>\glssymbolpluralaccessdisplay</code>	108	<code>\glxtr@s@newignoredglossary</code>	24
<code>\glstarget</code>	212–217, 227	<code>\glxtr@s@provideignoredglossary</code>	25
<code>\GLStext</code>	197	<code>\glxtr@saveentrycounter</code>	6, 7, 57
<code>\Glstext</code>	197	<code>\glxtr@setup@record</code>	8, 9, 16
<code>\glstext</code>	197	<code>\glxtr@shortcutsval</code>	100
<code>\glstextaccessdisplay</code>	105	<code>\glxtr@texencoding</code>	100
<code>\glstextformat</code>	41	<code>\glxtr@usesee</code>	29
<code>\glstextup</code>	170	<code>\glxtr@warnonexistsordo</code>	6, 9, 27, 28
<code>\glstreeindent</code>	226, 227	<code>\glxtr@writefields</code>	98
<code>\glstreenamebox</code>	227	<code>\glxtrabbrvfootnote</code>	162–164
<code>\glstreenamefmt</code>	218–227	<code>\glxtrabbrvpluralsuffix</code>	100, 143, 158, 160, 162, 164, 166, 167, 169, 170, 174, 185
<code>\GlstrelatField</code>	20	<code>\glxtrabbrvtype</code>	11, 140
<code>\glstype</code>	49–53, 143–152	<code>\glxtraddallcrossrefs</code>	29
<code>\glunset</code>	30, 73, 74	<code>\glxtralias</code>	57
<code>\glswrite</code>	85	<code>\glxtrAltTreeIndent</code>	218
<code>\glswriteentry</code>	6	<code>\glxtralttreeInit</code>	226
<code>\Glsxtr</code>	34	<code>\glxtrAltTreePar</code>	217
		<code>\glxtrAltTreeSetHangIndent</code>	218, 227
		<code>\glxtrAltTreeSetSubHangIndent</code>	227
		<code>\glxtralttreeSubSymbolDescLocation</code>	227
		<code>\glxtralttreeSymbolDescLocation</code>	218, 227

<code>\glxtrassignfieldfont</code>	42–49	<code>\glxtrheadfull</code>	194
<code>\glxtrcat</code>	32, 33	<code>\Glsxtrheadfullpl</code>	194
<code>\glxtrchecknohyperfirst</code>	43, 44	<code>\glxtrheadfullpl</code>	194
<code>\glxtrComputeTreeIndent</code>	227	<code>\Glsxtrheadlong</code>	194
<code>\glxtrComputeTreeSubIndent</code>	227	<code>\glxtrheadlong</code>	194
<code>\GlsXtrDefineAbbreviationShortcuts</code> .	13	<code>\Glsxtrheadlongpl</code>	194
<code>\GlsXtrDefineOtherShortcuts</code>	13	<code>\glxtrheadlongpl</code>	194
<code>\glxtrdiscardperiod</code>	134	<code>\Glsxtrheadplural</code>	194
<code>\glxtrdisplayendloc</code>	93	<code>\glxtrheadplural</code>	194
<code>\glxtrdisplayendloohook</code>	93	<code>\Glsxtrheadshort</code>	194
<code>\glxtrdisplaysingleloc</code>	93	<code>\glxtrheadshort</code>	194
<code>\glxtrdisplaystartloc</code>	93	<code>\Glsxtrheadshortpl</code>	194
<code>\glxtrdoautoindexname</code>	57, 58, 126	<code>\glxtrheadshortpl</code>	194
<code>\glxtrdopostpunc</code>	164	<code>\Glsxtrheadtext</code>	194
<code>\glxtrdowrglossaryhook</code>	58	<code>\glxtrheadtext</code>	194
<code>\GlsXtrEnableEntryCounting</code>	82	<code>\glxtrifcounttrigger</code>	73, 74
<code>\GlsXtrEnableEntryUnitCounting</code>	70	<code>\glxtrifemptyglossary</code>	92, 97, 101
<code>\GlsXtrEnableOnTheFly</code>	32, 34	<code>\glxtrifindexing</code>	57
<code>\glxtrfieldlistgadd</code>	100	<code>\glxtrifinmark</code>	66–69, 193, 194
<code>\glxtrfieldtitlecase</code>	121–124	<code>\glxtrifnextpunc</code>	136, 137
<code>\glxtrfieldtitlecasecs</code>	120	<code>\glxtrifperiod</code>	135, 136
<code>\glxtrfieldxifinlist</code>	102	<code>\glxtrifwasfirstuse</code> .	41, 43, 44, 49–53, 55, 84, 135, 143, 146–152, 164, 186, 188, 190
<code>\glxtrfirstscfont</code>	171–174	<code>\glxtrindexaliased</code>	56, 57
<code>\glxtrfirstsmfont</code>	174–177	<code>\Glsxtrinlinefullformat</code>	142, 144, 156, 163, 165, 166, 168, 169, 187, 189, 208
<code>\GlsXtrFmtDefaultOptions</code>	18	<code>\glxtrinlinefullformat</code>	142– 144, 156, 163, 165–167, 169, 187, 189, 208
<code>\GlsXtrFmtField</code>	18, 19	<code>\Glsxtrinlinefullplformat</code> .	142, 145, 156, 163, 165, 166, 168, 169, 187, 189, 209
<code>\GlsXtrFormatLocationList</code> 36, 38, 224–226		<code>\glxtrinlinefullplformat</code> .	141, 142, 145, 156, 163, 165–167, 169, 187, 189, 208
<code>\GLSxtrfull</code>	12, 202	<code>\glxtrininsertinsidefalse</code>	158
<code>\Glsxtrfull</code>	12, 202, 203	<code>\glxtrlocrangefmt</code>	93
<code>\glxtrfull</code>	12, 202	<code>\GLSxtrlong</code>	12, 200, 201
<code>\Glsxtrfullformat</code> 141, 154, 156, 159, 160, 163, 165, 166, 168, 169, 186, 187, 189, 191		<code>\Glsxtrlong</code>	12, 201
<code>\glxtrfullformat</code> .	141, 154, 156, 158– 162, 164, 166, 168, 169, 185, 187, 189, 191	<code>\glxtrlong</code>	11, 200
<code>\GLSxtrfullpl</code>	12, 202, 203	<code>\GLSxtrlongpl</code>	12, 200, 201
<code>\Glsxtrfullpl</code>	12, 203	<code>\Glsxtrlongpl</code>	12, 201
<code>\glxtrfullpl</code>	12, 202	<code>\glxtrlongpl</code>	12, 200
<code>\Glsxtrfullplformat</code> .	142, 154, 156, 159, 161, 163, 165, 167–169, 186, 187, 189, 191	<code>\glxtrlongshortdescsort</code>	159
<code>\glxtrfullplformat</code> .	154, 156, 158, 160, 162, 164, 166, 168, 169, 185, 187, 189, 191	<code>\glxtrmarkhook</code>	192, 193
<code>\glxtrfullsep</code>	141, 158–161, 163, 165–169, 178–180, 184	<code>\glxtrnewabbrevpresetkeyhook</code>	140
<code>\glxtrgenabbrvfmt</code>	39	<code>\glxtrnewnumber</code>	12
<code>\Glsxtrheadfirst</code>	194	<code>\glxtrnewsymbol</code>	12
<code>\glxtrheadfirst</code>	194	<code>\glxtrNoGlossaryWarning</code>	14, 94
<code>\Glsxtrheadfirstplural</code>	194	<code>\GlsXtrNoGlsWarningAutoMake</code>	97
<code>\glxtrheadfirstplural</code>	194	<code>\GlsXtrNoGlsWarningBuildInfo</code>	98
<code>\Glsxtrheadfull</code>	194	<code>\GlsXtrNoGlsWarningCheckFile</code>	97

<code>\GlsXtrNoGlsWarningEmptyMain</code>	97, 98	<code>\glxtrrmsuffix</code>	174–177
<code>\GlsXtrNoGlsWarningEmptyNotMain</code>	97	<code>\glxtrrtagfont</code>	133
<code>\GlsXtrNoGlsWarningEmptyStart</code>	97	<code>\Glsxtrrtitlefirst</code>	193, 194, 206
<code>\GlsXtrNoGlsWarningHead</code>	97	<code>\glxtrrtitlefirst</code>	193, 194, 206
<code>\GlsXtrNoGlsWarningMisMatch</code>	98	<code>\Glsxtrrtitlefirstplural</code>	193–195, 206, 207
<code>\GlsXtrNoGlsWarningNoOut</code>	98	<code>\glxtrrtitlefirstplural</code>	193–195, 206
<code>\GlsXtrNoGlsWarningTail</code>	98	<code>\Glsxtrrtitlefull</code>	194, 195, 208
<code>\glxtrorg@ifKV@glslink@hyper</code>	39	<code>\glxtrrtitlefull</code>	194, 195, 208
<code>\GLSxtrp</code>	65	<code>\Glsxtrrtitlefullpl</code>	194, 195, 209
<code>\Glsxtrp</code>	65	<code>\glxtrrtitlefullpl</code>	194, 195, 208, 209
<code>\glxtrp</code>	65, 67	<code>\Glsxtrrtitlelong</code>	194, 195, 207
<code>\Glsxtrpl</code>	34	<code>\glxtrrtitlelong</code>	193–195, 207
<code>\glxtrpl</code>	34	<code>\Glsxtrrtitlelongpl</code>	194, 195, 208
<code>\glxtrpostdescription</code>	119, 133, 217	<code>\glxtrrtitlelongpl</code>	193–195, 207
<code>\glxtrpostlink</code>	134	<code>\Glsxtrrtitleplural</code>	193, 194, 205
<code>\glxtrpostlinkendsentence</code>	134	<code>\glxtrrtitleplural</code>	193, 194, 205
<code>\glxtrpostlinkhook</code>	134	<code>\Glsxtrrtitleshort</code>	193, 194, 204
<code>\glxtrpostlocalreset</code>	70–72, 80	<code>\glxtrrtitleshort</code>	193, 194, 203
<code>\glxtrpostlocalunset</code>	69, 71, 79	<code>\Glsxtrrtitleshortpl</code>	193, 194, 204
<code>\glxtrpostlongdescription</code>	23	<code>\glxtrrtitleshortpl</code>	193, 194, 204
<code>\glxtrpostnamehook</code>	123–126	<code>\Glsxtrrtitletext</code>	193, 194, 205
<code>\GlsXtrPostNewAbbreviation</code>		<code>\glxtrrtitletext</code>	193, 194, 204, 205
....	140, 156, 158–162, 164, 166–168, 170, 179, 180, 182, 185, 186, 188, 190, 191	<code>\glxtrtreetopindent</code>	218, 226
<code>\glxtrpostreset</code>	70, 71, 79	<code>\glxtrundefaction</code>	6, 9, 17, 24, 26–28
<code>\glxtrpostunset</code>	69, 71, 79	<code>\glxtrundeftag</code>	16
<code>\glxtrprotectlinks</code>	59, 60	<code>\glxtrunsrtdo</code>	102
<code>\GlsXtrRecordCounter</code>	7	<code>\GlsXtrUseAbbrStyleFmts</code>	
<code>\glxtrregularfont</code>	39, 42	159, 161, 170–184, 188, 190, 192
<code>\glxtrresourcecount</code>	99	<code>\GlsXtrUseAbbrStyleSetup</code>	170–183, 190, 192
<code>\glxtrresourcefile</code>	99	<code>\glxtruserfield</code>	184
<code>\glxtrrestoremarkhook</code>	192, 193	<code>\glxtruserparen</code>	185–191
<code>\glxtrscfont</code>	170–174	<code>\glxtrusersuffix</code>	185, 186, 189, 191
<code>\glxtrscsuffix</code>	171–174	<code>\glxtruseseeformat</code>	29
<code>\GlsXtrSetActualChar</code>	130	<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	138, 157
<code>\glxtrsetaliasnoindex</code>	56, 57	<code>\GlsXtrWarning</code>	32, 33
<code>\GlsXtrSetEncapChar</code>	130		
<code>\GlsXtrSetEscChar</code>	130		
<code>\glxtrsetfieldifexists</code>	20, 21		
<code>\GlsXtrSetLevelChar</code>	130		
<code>\glxtrsetpopts</code>	64		
<code>\glxtrsetupfulldefs</code>	143–145, 164		
<code>\GLSxtrshort</code>	12, 68, 69, 195, 196		
<code>\Glsxtrshort</code>	12, 196		
<code>\glxtrshort</code>	11, 195		
<code>\GLSxtrshortpl</code>	12, 195, 196		
<code>\Glsxtrshortpl</code>	12, 196		
<code>\glxtrshortpl</code>	11, 195, 196		
<code>\glxtrrmsfont</code>	174–177		

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
<code>\hyperpage</code>	126
<code>\hyperref</code>	59
<code>hyperref package</code>	60, 126, 192, 203

I

<code>\if</code>	32
------------------------	----

makeglossaries	90	loclist	19
\makeglossaries	85, 95–98	long	111, 114, 207
\makeglossary	86	longplural	111, 114, 207
makeindex	229	name	104, 111, 112, 127
makeindex	85	plural	105, 106, 112, 157, 197, 198, 205
\makenoidxglossaries	96	see	10, 28, 31, 86
\MakeTextUppercase	194	short	110, 114, 138
\MakeUppercase	193, 194	shortplural	110, 114, 138
\markboth	193	symbol	107, 113
\markright	193	symbolplural	108, 113
\maxdimen	35, 36	text	59, 105, 112, 157, 159, 196, 197, 204
\mbox	227	\newif	126, 158
\medskip	97, 98, 102	\newlength	218
\MessageBreak ..	31, 34, 72, 73, 81, 88, 89, 155	\newnum	12
mfistuc package	132	\newrobustcmd ..	18–21, 54, 55, 64–66, 75, 132, 133, 143–152, 193, 195–203, 219–225
\mfistucMakeUppercase	42–53, 55, 62–64, 66, 68, 69, 75, 83, 105–114, 123, 124, 144, 145, 147, 149, 150, 152–154	\newsym	12
\mfu@checkword@arg	132	\newterm	118
\mfu@checkword@do	132	\newtoks	138
N		\newwrite	85
\NeedsTeXFormat	4, 211	\NoCaseChange	66–69, 195–203
\new@glossaryentry	31, 88	\noexpand 7, 14, 22, 83, 94, 98, 127, 128, 140, 211	
\new@ifnextchar	54, 55, 75, 136, 143–152	\nofiles	97
\newabbr	12	\noindent	97, 98
\newabbreviation	12	\nopostdesc	23, 32, 33, 89, 119
\newabbreviationhook	140	\nr	6, 8, 10, 13, 14, 90
\newabbreviationstyle	158–163, 165, 167, 168, 170–183, 185–188, 190, 192	\ns@GLSxtrfull	144
\newacronym	82, 84	\ns@Glsxtrfull	143
\newacronymhook	83	\ns@glxtrfull	143
\newacronymstyle	84	\ns@GLSxtrfullpl	145
\newcommand	4–12, 14–34, 36–39, 41, 42, 53–59, 61, 64, 66–71, 73, 75–79, 81, 82, 84, 85, 88, 90–120, 126– 152, 154–157, 159, 161, 162, 170, 174, 178, 184, 185, 193–209, 211, 217–219, 226	\ns@Glsxtrfullpl	145
\newcount	9, 99	\ns@glxtrfullpl	144
\newentry	12	\ns@GLSxtrlong	148
\newglossary	11, 86	\ns@Glsxtrlong	148
\newglossaryentry	12, 31, 71, 78, 83, 118, 119, 140	\ns@glxtrlong	147
\newglossaryentry options		\ns@GLSxtrlongpl	152
alias	22	\ns@Glsxtrlongpl	151
desc	108, 109, 113	\ns@glxtrlongpl	151
descplural	109, 113, 114	\ns@GLSxtrshort	147
first	59, 106, 112, 157, 198–200, 205, 229	\ns@Glsxtrshort	146
firstplural	106, 107, 112, 113, 157, 199, 206, 229	\ns@glxtrshort	146
		\ns@GLSxtrshortpl	150
		\ns@Glsxtrshortpl	149, 150
		\ns@glxtrshortpl	149
		\null	14
		\number	77–80, 98
		\numexpr	77, 78, 80
O		\or	6, 9, 13, 31

\org@glossaryentrynumbers	36, 89	\printnoidxglossaries	96
\org@glossarytitle	89	\printnoidxglossary	87, 96
P			
\p@glshyp@opt	58	\printnumbers	12, 119
package options:		\printsymbols	12, 119
abbreviations	11	\printunsrtglossary	101
acccsupp	14, 104	\printunsrtglossaryhandler	102
acronym	11	\printunsrtglossaryunit	9, 102
automake	88, 96	\printunsrtglossaryunitsetup	102
docdef	9, 30, 71, 78	\ProcessOptions	211
false	31	\ProcessOptionsX	15
restricted	10	\protect	66–69, 112–114, 141, 158–180, 182, 185–188, 190, 191, 195–203
true	31	\protected@csedef	21, 218
nonumberlist	36	\protected@csxdef	21, 218
nopostdot		\protected@edef	35, 83, 127, 140
false	11	\protected@write	7, 38, 85, 86, 98–101
numbers	12	\protected@xdef	103
record	6, 8, 39, 98, 238	\providecommand	11, 17, 38, 55, 72, 81, 85, 94, 211
shortcuts	13	\ProvidesFile	209
all	13	\ProvidesPackage	4, 211
false	13	Q	
none	13	\quotechar	130
true	13	R	
style	15	\raggedright	214, 216
stylemods	15	\relax	6, 8– 10, 12–16, 18, 31, 34–37, 41, 58, 64, 72, 73, 80, 81, 86, 89, 91, 93, 98–100, 103, 128, 130, 132, 134, 135, 138, 139, 219–228
symbols	12, 119	resize package	174
undefaction	27	\renewcommand	5, 9– 11, 13–15, 22–28, 30, 31, 34–39, 41, 53, 55–57, 59, 60, 64, 69–72, 76, 78–80, 82– 89, 92–95, 102, 118, 120–123, 125, 130– 134, 142, 156, 158–193, 212–217, 227, 228
error	5	\renewenvironment	212–216, 226
warn	5	\renewglossarystyle	212–216, 226
\PackageError	5, 7, 14, 31, 34, 35, 54, 55, 57, 65, 70–72, 78, 80, 82, 84, 86–88, 92, 102, 155–157, 211	\renewrobustcmd	41
\PackageWarning	10	\RequireGlossariesExtraLang	210
\PackageWarningNoLine	10	\RequirePackage	4, 14, 15, 211
\pageref	21	\reserved@a	136
\par	97, 98, 217, 218, 227	\reserved@b	136
\parindent	218, 227	\reserved@d	137
\PassOptionsToPackage	4	\RestoreAcronyms	84
\preglossarypreamble	22	\romannumeral	218, 219, 226
\preto	56	S	
\print@noop@unsrtglossaryunit	7, 9	\s@glshyp@opt	58
\print@op@unsrtglossaryunit	9		
\printabbreviations	11		
\printglossaries	86, 96		
\printglossary	11, 86, 96		
\printglossary options			
nonumberlist	38		
type	88		

