

glossaries-extra.sty v1.11: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2017-01-19

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	16
1.3.1	Existence Checks	17
1.3.2	Document Definitions	20
1.3.3	Existing Glossary Style Modifications	25
1.3.4	Entry Formatting, Hyperlinks and Indexing	29
1.3.5	Entry Counting	58
1.3.6	Acronym Modifications	72
1.3.7	Indexing and Displaying Glossaries	74
1.4	Integration with glossaries-accsupp	89
1.5	Categories	99
1.6	Abbreviations	122
1.6.1	Abbreviation Styles Setup	139
1.6.2	Predefined Styles (Default Font)	142
1.6.3	Predefined Styles (Small Capitals)	154
1.6.4	Predefined Styles (Fake Small Capitals)	158
1.6.5	Predefined Styles (Emphasized)	162
1.6.6	Predefined Styles (User Parentheses Hook)	168
1.7	Using Entries in Headings	172
1.8	Multi-Lingual Support	189
2	Style Adjustments (glossaries-extra-stylemods.sty)	191
2.1	Package Initialisation	191
2.2	List-Like Styles	192
2.3	Longtable Styles	192
2.4	Long Ragged Styles	193
2.5	Supertabular Styles	195
2.6	Super Ragged Styles	196
2.7	Inline Style	197
2.8	Tree Styles	197
	Glossary	209
	Change History	210
	Index	219

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2017/01/19 v1.11 (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\@glstr@warn@undefaction
68     \let\glstr@warnonexistssordo\@glstr@warn@onexistssordo
69     \let\@glstr@redef@forglsentries\@glstr@do@redef@forglsentries
70   \or
71     \let\glstrundefaction\@glstr@err@undefaction
72     \let\glstr@warnonexistssordo\@gobble
73     \let\@glstr@redef@forglsentries\relax
74   \fi
75 }

```

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 \glxtr@@do@@wrglossary.

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

glxtr@dorecord

```
99 \newcommand*\glxtr@dorecord{%
100   \protected@write\@auxout{}\string\glxtr@record
101     {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
102     {\@glslocref}}%
103 }
```

\glxtr@record

```
104 \newcommand*\glxtr@record[5]{}
```

tr@setup@record Initialise.

```
105 \newcommand*\glxtr@setup@record{}
```

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

```
106 \newcommand*\glxtr@indexonly@saveentrycounter{%
107   \ifKV@glslink@noindex
108   \else
109     \glxtr@saveentrycounter
110   \fi
111 }
```

addloclistfield

```
112 \newcommand*\glxtr@addloclistfield{%
113   \key@ifundefined{glossentry}{loclist}%
114   {%
115     \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
116     \appto\@gls@keymap{,loclistloclist}%
117     \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
118     \appto\@newglossaryentryposthook{%
119       \gls@assign@field{\@glo@label}{loclist}{\@glo@loclist}%
120     }%
121     \glssetnoexpandfield{loclist}%
122   }%
123   {}}%
```

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

```
124 \key@ifundefined{glossentry}{location}%
125 {%
126   \define@key{glossentry}{location}{\def\@glo@location{##1}}%
127   \appto\@gls@keymap{,locationlocation}%
128   \appto\@newglossaryentryprehook{\def\@glo@location{}}%
```

```

129 \appto\@newglossaryentryposthook{%
130   \gls@assign@field{}\@glo@label}{location}{\@glo@location}%
131 }%
132 \glssetnoexpandfield{location}%
133 }%
134 {}%

```

Add a key to store the group heading.

```

135 \key@ifundefined{glossentry}{group}%
136 {%
137   \define@key{glossentry}{group}{\def\@glo@group{##1}}%
138   \appto\@gls@keymap{,{group}{group}}%
139   \appto\@newglossaryentryprehook{\def\@glo@group{}}%
140   \appto\@newglossaryentryposthook{%
141     \gls@assign@field{}\@glo@label}{group}{\@glo@group}%
142   }%
143   \glssetnoexpandfield{group}%
144 }%
145 {}%
146 }

```

Now define the record package option.

```

147 \define@choicekey{glossaries-extra.sty}{record}[\val\nr]%
148 {off,only,alsoindex}%
149 [only]%
150 {%
151   \ifcase\nr\relax

```

Don't record.

```

152   \def\glsxtr@setup@record{%
153     \renewcommand*{\@glsxtr@record}[2]{}%
154     \let\@do@wrglossary\glsxtr@do@wrglossary
155     \let\@gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
156     \let\glsxtrundefaction\@glsxtr@err@undefaction
157     \let\glsxtr@warnonexistsordo\@gobble
158   }%
159   \or

```

Only record (don't index).

```

160   \def\glsxtr@setup@record{%
161     \let\@glsxtr@record\@glsxtr@record
162     \let\@do@wrglossary\@gobble
163     \let\@gls@saveentrycounter\relax
164     \let\glsxtrundefaction\@glsxtr@warn@undefaction
165     \let\glsxtr@warnonexistsordo\@glsxtr@warn@onexistsordo
166     \glsxtr@addloclistfield
167   }%
168   \or

```

Record and index.

```

169   \def\glsxtr@setup@record{%

```

```

170      \let\@glstr@record\@glstr@record
171      \let\@do@wrglossary\glstr@do@wrglossary
172      \let\@gl@saveentrycounter\glstr@indexonly@saveentrycounter
173      \let\glstrundefaction\glstr@warn@undefaction
174      \let\glstr@warnonexistsordo\glstr@warn@onexistsordo
175      \glstr@addloclistfield
176    }%
177  \fi
178 }

```

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

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

```

179 \newcount\@glstr@docdefval

```

Need to provide conditional commands that are backward compatible:

```

if@glstrdocdef
180 \newcommand*\if@glstrdocdef{\ifnum\@glstr@docdefval>0 }

glstrdocdeftrue
181 \newcommand*\@glstrdocdeftrue{\@glstr@docdefval=1 }

glstrdocdeffalse
182 \newcommand*\@glstrdocdeffalse{\@glstr@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.

```

183 \define@choicekey{glossaries-extra.sty}{docdef}[\val\nr]%
184 {false,true,restricted}[true]%
185 {%
186   \@glstr@docdefval=\nr\relax
187   \ifnum\@glstr@docdefval=2\relax
188     \renewcommand*\@glsdofexistsorwarn{\glsdofexists}%
189   \fi
190 }

```

docdefrestricted

```

191 \newcommand*\if@glstrdocdefrestricted{\ifnum\@glstr@docdefval=2 }

```

glsdofexistsorwarn 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).

```

192 \newcommand*\@glsdofexistsorwarn{\glsdofexistsorwarn}

```

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

```

193 \define@boolkey{glossaries-extra.sty}[@glxtr]{indexcrossrefs}[true]{%
194 \if@glxtrindexcrossrefs
195 \else
196 \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
197 \fi
198 }

Switch off since this can increase the build time.
199 \@glxtrindexcrossrefsfalse

But allow see key to switch it on automatically.

oindexcrossrefs
200 \newcommand*{\@glxtr@autoindexcrossrefs}{\@glxtrindexcrossrefstrue}

iesExtraWarning Allow users to suppress warnings.
201 \newcommand*{\GlossariesExtraWarning}[1]{\PackageWarning{glossaries-extra}{#1}}

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

204 \@glxtr@declareoption{nowarn}{%
205 \let\GlossariesExtraWarning\@gobble
206 \let\GlossariesExtraWarningNoLine\@gobble
207 \glxtr@doption{nowarn}%
208 }

glxtrabbrvtype Glossary type for abbreviations.
209 \newcommand*{\glxtrabbrvtype}{\glstdefaulttype}

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

bbreviationsdef
211 \newcommand*{\@glxtr@doabbreviationsdef}{%
212 \@ifpackageloaded{babel}%
213 {\providecommand{\abbreviationsname}{\acronymname}}%
214 {\providecommand{\abbreviationsname}{Abbreviations}}%
215 \newglossary[glg-abr]{abbreviations}{glg-abr}{glo-abr}{\abbreviationsname}%
216 \renewcommand*{\glxtrabbrvtype}{abbreviations}%
217 \newcommand*{\printabbreviations}[1][1]{%
218 \printglossary[type=\glxtrabbrvtype,##1]%
219 }%
220 \disable@keys{glossaries-extra.sty}{abbreviations}%

```

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

```
221 \ifglssacronym
222 \else
223   \renewcommand*{\acronymtype}{\glstrabbrvtype}%
224 \fi
225 }%
```

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

```
226 \@glstr@declareoption{abbreviations}{%
227   \let\@glstr@abbreviationsdef\@glstr@doabbreviationsdef
228 }
```

AbbreviationShortcuts Enable shortcut commands for the abbreviations. Unlike the analogous command provided by glossaries, this uses `\newcommand` instead of `\let` as a safety feature.

```
229 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
230   \newcommand*{\ab}{\cglS}%
231   \newcommand*{\abp}{\cglSpl}%
232   \newcommand*{\as}{\glstrshort}%
233   \newcommand*{\asp}{\glstrshortpl}%
234   \newcommand*{\al}{\glstrlong}%
235   \newcommand*{\alp}{\glstrlongpl}%
236   \newcommand*{\af}{\glstrfull}%
237   \newcommand*{\afp}{\glstrfullpl}%
238   \newcommand*{\Ab}{\cGlS}%
239   \newcommand*{\Abp}{\cGlSpl}%
240   \newcommand*{\As}{\GlsXtrshort}%
241   \newcommand*{\Asp}{\GlsXtrshortpl}%
242   \newcommand*{\Al}{\GlsXtrlong}%
243   \newcommand*{\Alp}{\GlsXtrlongpl}%
244   \newcommand*{\Af}{\GlsXtrfull}%
245   \newcommand*{\Afp}{\GlsXtrfullpl}%
246   \newcommand*{\AB}{\cGLS}%
247   \newcommand*{\ABP}{\cGLSpl}%
248   \newcommand*{\AS}{\GLSxtrshort}%
249   \newcommand*{\ASP}{\GLSxtrshortpl}%
250   \newcommand*{\AL}{\GLSxtrlong}%
251   \newcommand*{\ALP}{\GLSxtrlongpl}%
252   \newcommand*{\AF}{\GLSxtrfull}%
253   \newcommand*{\AFP}{\GLSxtrfullpl}%
254   \newcommand*{\newabbr}{\newabbreviation}%

```

Disable this command after it's been used.

```
255   \let\GlsXtrDefineAbbreviationShortcuts\relax
256 }
```

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

```
257 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
258   \newcommand*{\newentry}{\newglossaryentry}%

```

```

259 \ifdef\printsymbols
260 {%
261   \newcommand*{\newsym}{\glxtrnewsymbol}%
262 }{}%
263 \ifdef\printnumbers
264 {%
265   \newcommand*{\newnum}{\glxtrnewnumber}%
266 }{}%
267 \let\GlsXtrDefineOtherShortcuts\relax
268 }

```

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.

```

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

```

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

```

270 \newcommand*{\@glxtr@shortcutsval}{\ifglssacrshortcutsacro\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.

```

271 \define@choicekey{glossaries-extra.sty}{shortcuts}[\val\nr]%
272 {acronyms,acro,abbreviations,abbr,other,all,true,none,false}[true]{%
273   \let\@glxtr@shortcutsval\val
274   \ifcase\nr\relax % acronyms
275     \renewcommand*{\@glxtr@setupshortcuts}{%
276       \glssacrshortcutstrue
277       \DefineAcronymSynonyms
278     }%
279   \or % acro
280     \renewcommand*{\@glxtr@setupshortcuts}{%
281       \glssacrshortcutstrue
282       \DefineAcronymSynonyms
283     }%
284   \or % abbreviations
285     \renewcommand*{\@glxtr@setupshortcuts}{%
286       \GlsXtrDefineAbbreviationShortcuts
287     }%
288   \or % abbr
289     \renewcommand*{\@glxtr@setupshortcuts}{%
290       \GlsXtrDefineAbbreviationShortcuts
291     }%
292   \or % other
293     \renewcommand*{\@glxtr@setupshortcuts}{%
294       \GlsXtrDefineOtherShortcuts

```

```

295     }%
296 \or % all
297   \renewcommand*{\@glxtr@setupshortcuts}{%
298     \glsacrshortcutstrue
299     \DefineAcronymSynonyms
300     \GlsXtrDefineAbbreviationShortcuts
301     \GlsXtrDefineOtherShortcuts
302   }%
303 \or % true
304   \renewcommand*{\@glxtr@setupshortcuts}{%
305     \glsacrshortcutstrue
306     \DefineAcronymSynonyms
307     \GlsXtrDefineAbbreviationShortcuts
308     \GlsXtrDefineOtherShortcuts
309   }%
310 \else % none, false
311   \renewcommand*{\@glxtr@setupshortcuts}{}%
312 \fi
313 }

```

`lsxtr@doaccsupp`

```

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

```

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

```

315 \@glxtr@declareoption{accsupp}{%
316 \renewcommand*{\@glxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}

```

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

```

317 \newcommand{\glxtrNoGlossaryWarning}[1]{%
318 \@glxtr@defaultnoglossarywarning{#1}%
319 }

```

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

```

320 \define@choicekey{glossaries-extra.sty}{nomissingglstext}[\val\nr]%
321 {true,false}[true]{%
322   \ifcase\nr\relax % true
323     \renewcommand{\glxtrNoGlossaryWarning}[1]{%
324       \null
325     }%
326   \else % false
327     \renewcommand{\glxtrNoGlossaryWarning}[1]{%
328       \@glxtr@defaultnoglossarywarning{#1}%
329     }%
330   \fi
331 }

```

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

xtr@redefstyles

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

stylemods

```
333 \define@key{glossaries-extra.sty}{stylemods}{%
334   \ifblank{#1}%
335   {%
336     \renewcommand*{\@glxtr@redefstyles}{%
337       \RequirePackage{glossaries-extra-stylemods}}%
338   }%
339   {%
340     \renewcommand*{\@glxtr@redefstyles}{}%
341     \@for\@glxtr@tmp:=#1\do{%
342       \IfFileExists{glossary-\@glxtr@tmp.sty}%
343       {%
344         \eappto\@glxtr@redefstyles{%
345           \noexpand\RequirePackage{glossary-\@glxtr@tmp}}%
346       }%
347       {%
348         \PackageError{glossaries-extra}%
349           {Glossaries style package ‘glossary-\@glxtr@tmp.sty’
350            doesn’t exist (did you mean to use the ‘style’ key?)}%
351           {The list of values (#1) in the ‘stylemods’ key should
352            match the glossary-xxx.sty files provided with
353            glossaries.sty}%
354       }%
355     }%
356     \appto\@glxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
357   }%
358 }
```

glxtr@do@style

```
359 \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.

```
360 \define@key{glossaries-extra.sty}{style}{%
361   \renewcommand*{\@glxtr@do@style}{%
```

Set this as the default style:

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

Set this style:

```
363   \setglossarystyle{#1}%
364 }%
365 }
```

Pass all other options to glossaries.

```
366 \DeclareOptionX*{%
367   \expandafter\glxtr@doooption\expandafter{\CurrentOption}}
```

Process options.

```

368 \ProcessOptionsX
    Load glossaries if not already loaded.
369 \RequirePackage{glossaries}
    Load the glossaries-accsupp package if required.
370 \@glsxtr@doaccsupp
    Define abbreviations glossaries if required.
371 \@glsxtr@abbreviationsdef
372 \let\@glsxtr@abbreviationsdef\relax
    Setup shortcuts if required.
373 \@glsxtr@setupshortcuts
    Redefine \@glsxtr@redef@forlgsentries if required.
374 \@glsxtr@redef@forlgsentries

```

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

```

375 \renewcommand{\@glsxtr@doption}[1]{\setupglossaries{#1}}%
    Now define the user command:
376 \newcommand*{\glossariesextrasetup}[1]{%
377   \let\@glsxtr@setup@record\relax
378   \let\@glsxtr@setupshortcuts\relax
379   \let\@glsxtr@redef@forlgsentries\relax
380   \setkeys{glossaries-extra.sty}{#1}%
381   \@glsxtr@abbreviationsdef
382   \let\@glsxtr@abbreviationsdef\relax
383   \@glsxtr@setupshortcuts
384   \glsxtr@setup@record
385   \@glsxtr@redef@forlgsentries
386 }

```

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

```

387 \let\@glsxtr@@do@wrglossary\@@do@wrglossary

```

saveentrycounter Save original definition of \@gls@saveentrycounter.

```

388 \let\@glsxtr@saveentrycounter\@gls@saveentrycounter

```

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

```

389 \let\@gls@saveentrycounter\@glsxtr@indexonly@saveentrycounter

```

Set up record option if required.

```

390 \glsxtr@setup@record

```

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

```
391 \AtBeginDocument{%
392   \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
393   \def\@glxstrundeftag{\glxstrundeftag}%
394 }
```

1.2 Extra Utilities

`\glxstrifemptyglossary`

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

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

```
395 \newcommand{\glxstrifemptyglossary}[3]{%
396   \ifglossaryexists{#1}%
397   {%
398     \ifcsstring{glolist@#1}{,}{#2}{#3}%
399   }%
400   {%
401     \glxstrundefaction{Glossary type '#1' doesn't exist}{}%
402     #2%
403   }%
404 }
```

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

```
405 \ifglentrycounter
406   \newcommand*{\glxstrpageref}[1]{\pageref{glentry-\glsdetoklabel{#1}}}
407 \else
408   \ifglssubentrycounter
409     \newcommand*{\glxstrpageref}[1]{\pageref{glentry-\glsdetoklabel{#1}}}
410   \else
411     \newcommand*{\glxstrpageref}[1]{\gls{#1}}
412   \fi
413 \fi
```

1.3 Modifications to Commands Provided by glossaries

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

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

ignoredglossary Save the original definition.

```
414 \let\glxtr@org@newignoredglossary\newignoredglossary
```

ignoredglossary Starred form.

```
415 \newcommand*{\glxtr@s@newignoredglossary}[1]{%
416   \ifdefempty\@ignored@glossaries
417   {%
418     \edef\@ignored@glossaries{#1}%
419   }%
420   {%
421     \eappto\@ignored@glossaries{,#1}%
422   }%
423   \csgdef{glolist@#1}{,}%
424   \ifcsundef{gls@#1@entryfmt}%
425   {%
426     \defglsentryfmt[#1]{\glsentryfmt}%
427   }%
428   {}%
429 }
```

ignoredglossary Redefine to check for star.

```
430 \renewcommand{\newignoredglossary}{%
431   \@ifstar\glxtr@s@newignoredglossary\glxtr@org@newignoredglossary
432 }
```

1.3.1 Existence Checks

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

```
433 \renewcommand{\glsdoifexists}[2]{%
434   \ifglsentryexists{#1}{#2}%
435   {%
```

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

```
436   \edef\glslabel{\glsdetoklabel{#1}}%
437   \glxtrundefaction{Glossary entry '\glslabel'
438     has not been defined}{You need to define a glossary entry before
439     you can reference it.}%
440   }%
441 }
```

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

```
442 \renewcommand{\glsdoifnoexists}[2]{%
443   \ifglsentryexists{#1}{%
444     \glxtrundefaction{Glossary entry '\glsdetoklabel{#1}'
445       has already been defined}{}}{#2}%
446 }
```

`\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.

```

447 \ifdef\glsdoifexistsordo
448 {%
449   \renewcommand{\glsdoifexistsordo}[3]{%
450     \ifglstryexists{#1}{#2}%
451     {%
452       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
453       has not been defined}{You need to define a glossary entry
454       before you can use it.}%
455       #3%
456     }%
457   }%
458 }
459 {%
460   \glstr@warnonexistsordo\glsdoifexistsordo
461   \newcommand{\glsdoifexistsordo}[3]{%
462     \ifglstryexists{#1}{#2}%
463     {%
464       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
465       has not been defined}{You need to define a glossary entry
466       before you can use it.}%
467       #3%
468     }%
469   }%
470 }

```

`\glsarynoexistsordo` Similarly for `\doifglossarynoexistsordo`.

```

471 \ifdef\doifglossarynoexistsordo
472 {%
473   \renewcommand{\doifglossarynoexistsordo}[3]{%
474     \ifglossaryexists{#1}%
475     {%
476       \glstrundefaction{Glossary type '#1' already exists}{}%
477       #3%
478     }%
479     {#2}%
480   }%
481 }
482 {%
483   \glstr@warnonexistsordo\doifglossarynoexistsordo
484   \newcommand{\doifglossarynoexistsordo}[3]{%
485     \ifglossaryexists{#1}%
486     {%
487       \glstrundefaction{Glossary type '#1' already exists}{}%
488       #3%
489     }%
490     {#2}%
491   }%

```

```

492 }
493
ryentryposthook Hook into end of \newglossaryentry to add “see” value as a field.
494 \appto\@newglossaryentryposthook{%
495   \ifdefvoid\@glo@see
496     {\csxdef{glo@\@glo@label @see}{}}}%
497   {%
498     \csxdef{glo@\@glo@label @see}{\@glo@see}%
499     \@glxtr@autoindexcrossrefs
500   }%
501 }
502 \appto\@gls@keymap{,{see}{see}}

\glxtrusesee Apply \glsseeformat to the see key if not empty.
503 \newcommand*{\glxtrusesee}[1]{%
504   \glsdoifexists{#1}%
505   {%
506     \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
507     \ifdefempty\@glo@see
508       {}%
509       {%
510         \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
511       }%
512     }%
513 }

\glxtr@usesee
514 \newcommand*{\glxtr@usesee}[1][\@seenname]{%
515   \@glxtr@usesee[#1]%
516 }

\@glxtr@usesee
517 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
518   \glxtruseseeformat{#1}{#2}%
519 }

xtruseseeformat The format used by \glxtrusesee. The first argument is the tag (such as \@seenname). The
second argument is the comma-separated list of cross-referenced labels.
520 \newcommand*{\glxtruseseeformat}[2]{%
521   \glsseeformat[#1]{#2}{}%
522 }

Add all unused cross-references at the end of the document.
523 \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.

```

```

524 \newcommand*{\glxtraddallcrossrefs}{%
525   \forallglossaries{\@glo@type}%
526   {%
527     \forallglsentries[\@glo@type]{\@glo@label}%
528     {%
529       \ifglsused{\@glo@label}{\@glxtr@addunusedxrefs{\@glo@label}}{}%
530     }%
531   }%
532 }

```

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

```

533 \newcommand*{\@glxtr@addunusedxrefs}[1]{%
534   \letcs{\@glo@see}{glo@glstdetoklabel{#1}@see}%
535   \ifdefvoid\@glo@see
536   {}%
537   {%
538     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
539   }%
540 }

```

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

```

541 \newcommand*{\glxtr@addunused}[1][]{%
542   \@glxtr@addunused
543 }

```

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

```

544 \def\@glxtr@addunused#1\@end@glxtr@addunused{%
545   \@for\@glxtr@label:=#1\do
546   {%
547     \ifglsused{\@glxtr@label}{}%
548     {%
549       \glsadd[format=glxtrunusedformat]{\@glxtr@label}%
550       \glsunset{\@glxtr@label}%
551       \@glxtr@addunusedxrefs{\@glxtr@label}%
552     }%
553   }%
554 }

```

xtrunusedformat

```

555 \newcommand*{\glxtrunusedformat}[1]{\unskip}

```

1.3.2 Document Definitions

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

```

556 \let\glxtr@orgmakenoidxglossaries\makenoidxglossaries
557 \renewcommand{\makenoidxglossaries}{%
558   \glxtr@orgmakenoidxglossaries
559   \if@glxtrdocdefrestricted

```

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

```

560 \renewcommand*\@gls@reference}[3]{%
561 \ifcsundef{@glsref@##1}{\csgdef{@glsref@##1}{}}{}%
562 \ifinlistcs{##2}{@glsref@##1}%
563 {}%
564 {\listcsgadd{@glsref@##1}{##2}}%
565 \ifcsundef{glo@glstdetoklabel{##2}@loclist}%
566 {\csgdef{glo@glstdetoklabel{##2}@loclist}{}}%
567 {}%
568 \listcsgadd{glo@glstdetoklabel{##2}@loclist}{##3}%
569 }%
570 \else

```

Disable document definitions.

```

571 \@glxtrdocdeffalse
572 \fi
573 \disable@keys{glossaries-extra.sty}{docdef}%
574 }

```

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

```

575 \renewcommand*\@gls@defdocnewglossaryentry{%
576 \ifcase\@glxtr@docdefval
    docdef=false:
577 \renewcommand*\@newglossaryentry}[2]{%
578 \PackageError{glossaries-extra}{Glossary entries must
579 be \MessageBreak defined in the preamble with \MessageBreak
580 package option 'docdef=false'\MessageBreak(consider using
581 'docdef=restricted')}{Move your glossary definitions to
582 the preamble. You can also put them in a \MessageBreak separate file
583 and load them with \string\loadglsentries.}%
584 }%
585 \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.

```

586 \let\gls@checkseeallowed\relax
587 \let\newglossaryentry\new@glossaryentry
588 \or

```

Restricted mode just needs to allow the see value.

```

589 \let\gls@checkseeallowed\relax
590 \fi
591 }%

```

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

```
592 \newcommand*{\GlsXtrEnableOnTheFly}{%
593   \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
594 }
```

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.)

```
595 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
596   \renewcommand*{\glsdetoklabel}[1]{%
597     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
598     {%
599       \expandafter\detokenize\expandafter{##1}%
600     }%
601     {\detokenize{##1}}%
602   }%
603   \@GlsXtrEnableOnTheFly
604 }
605 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
606   \expandafter\if\glsbackslash#1%
607     #3%
608   \else
609     #4%
610   \fi
611 }
```

sxtrstarflywarn

```
612 \newcommand*{\glsxtrstarflywarn}{%
613   \GlossariesExtraWarning{Experimental starred version of
614   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
615   read the warnings in the glossaries-extra user manual)}%
616 }
```

rEnableOnTheFly

```
617 \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

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

\glsxtr

```
619 \newcommand*{\glsxtr}[1][]{%
620   \def\glsxtr@keylist{##1}%
621   \@glsxtr
622 }
```

```

\@glxstr
623 \newcommand*{\@glxstr}[2] [] {%
624 \ifglentryexists{##2}%
625 {%
626 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
627 }%
628 {%
629 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
630 description={\nopostdesc},##1}%
631 }%
632 \expandafter\gls\expandafter[\glxstr@keylist]{##2}%
633 }

\Glsxtr
634 \newcommand*{\Glsxtr}[1] [] {%
635 \def\glxtr@keylist{##1}%
636 \@Glsxtr
637 }

\@Glsxtr
638 \newcommand*{\@Glsxtr}[2] [] {%
639 \ifglentryexists{##2}%
640 {%
641 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
642 }%
643 {%
644 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
645 description={\nopostdesc},##1}%
646 }%
647 \expandafter\gls\expandafter[\glxtr@keylist]{##2}%
648 }

\glxtrpl
649 \newcommand*{\glxtrpl}[1] [] {%
650 \def\glxtr@keylist{##1}%
651 \@glxtrpl
652 }

\@glxtrpl
653 \newcommand*{\@glxtrpl}[2] [] {%
654 \ifglentryexists{##2}%
655 {%
656 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
657 }%
658 {%
659 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
660 description={\nopostdesc},##1}%
661 }%
662 \expandafter\glspl\expandafter[\glxtr@keylist]{##2}%

```

```
663 }
```

```
\Glsxtrpl
```

```
664 \newcommand*\Glsxtrpl[1][]{%
665 \def\glxtr@keylist{##1}%
666 \@Glsxtrpl
667 }
```

```
\@Glsxtrpl
```

```
668 \newcommand*\@Glsxtrpl[2][]{%
669 \ifglstryexists{##2}
670 {%
671 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
672 }%
673 {%
674 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
675 description={\nopostdesc},##1}%
676 }%
677 \expandafter\Glspl\expandafter[\glxtr@keylist]{##2}%
678 }
```

```
\GlsXtrWarning
```

```
679 \newcommand*\GlsXtrWarning[2]{%
680 \def\@glxtr@optlist{##1}%
681 \@onelevel@sanitize\@glxtr@optlist
682 \GlossariesExtraWarning{The options '\@glxtr@optlist' have
683 been ignored for entry '##2' as it has already been defined}%
684 }
```

Disable commands after the glossary:

```
685 \renewcommand\@printglossary[2]{%
686 \def\@glxtr@printglossopts{##1}%
687 \@glxtr@orgprintglossary{##1}{##2}%
688 \def\@glxtr{\@glxtr@disabledflycommand\glxtr}%
689 \def\@glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
690 \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
691 \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
692 }
```

```
abledflycommand
```

```
693 \newcommand*\@glxtr@disabledflycommand[1]{%
694 \PackageError{glossaries-extra}%
695 {\string##1\space can't be used after any of the \MessageBreak
696 glossaries have been displayed}%
697 {The on-the-fly commands enabled by
698 \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
699 before the glossaries. If you want to use any entries \MessageBreak
700 after any of the glossaries, you must use the standard \MessageBreak
701 method of first defining the entry and then using the \MessageBreak
```

```

702     entry with commands like \string\gls}%
703     \@@glstr@disabledflycommand
704 }%
705 \newcommand*{\@@glstr@disabledflycommand}[2][\{##2}

```

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

```

706 \let\GlsXtrEnableOnTheFly\relax
707 }
708 \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.

`\currentstyle` Initialise the current style to the default style.

```

709 \newcommand*{\@glstr@current@style}{\@glossary@default@style}

```

Modify \setglossarystyle to set the above.

`\setglossarystyle`

```

710 \renewcommand*{\setglossarystyle}[1]{%
711   \ifcsundef{@glsstyle@#1}%
712   {%
713     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
714   }%
715   {%
716     \csname @glsstyle@#1\endcsname
717   }%
718 }%
719 \ifx\@glossary@default@style\relax
720   \protected@edef\@glossary@default@style{#1}%
721 \fi
722 }

```

In case we have an old version of glossaries:

```

723 \ifdef\@glossary@default@style
724 {}
725 {%
726   \let\@glossary@default@style\relax
727 }

```

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

```

728 \ifdef\glslistdottedwidth
729 {%
730   \ifdim\glslistdottedwidth=.5\hsize

```

```

731 \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
732 \AtBeginDocument{%
733 \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
734 \setlength{\glslistdottedwidth}{.5\columnwidth}%
735 \fi
736 }%
737 \fi
738 }
739 {}%

```

Similarly for \glsdescwidth:

\glsdescwidth

```

740 \ifdef\glsdescwidth
741 {%
742 \ifdim\glsdescwidth=.6\hsize
743 \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
744 \AtBeginDocument{%
745 \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
746 \setlength{\glsdescwidth}{.6\columnwidth}%
747 \fi
748 }%
749 \fi
750 }
751 {}%

```

and for \glspagelistwidth:

\glspagelistwidth

```

752 \ifdef\glspagelistwidth
753 {%
754 \ifdim\glspagelistwidth=.1\hsize
755 \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
756 \AtBeginDocument{%
757 \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
758 \setlength{\glspagelistwidth}{.1\columnwidth}%
759 \fi
760 }%
761 \fi
762 }
763 {}%

```

aryentrynumbers Has the nonumberlist option been used?

```

764 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
765 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
766 \glsnonumberlistfalse
767 \renewcommand*{\glossaryentrynumbers}[1]{%
768 \ifglstryentryexists{\glscurrententrylabel}%
769 {}%
770 \@glsxtrpreloctag

```

```

771     \GlsXtrFormatLocationList{#1}%
772     \@glstrpostloctag
773     \gls@save@numberlist{#1}%
774   }{}%
775 }%
776 \else
777   \glsnonumberlisttrue
778   \renewcommand*{\glossaryentrynumbers}[1]{%
779     \ifglentryexists{\glscurrententrylabel}%
780     {%
781       \gls@save@numberlist{#1}%
782     }{}%
783   }%
784 \fi

```

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

```

785 \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`

```

786 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
787   \let\@glstrpreloctag\@glstrpreloctag
788   \let\@glstrpostloctag\@glstrpostloctag
789   \renewcommand*{\@glstr@pagetag}{#1}%
790   \renewcommand*{\@glstr@pagetag}{#2}%
791   \renewcommand*{\@glstr@savepreloctag}[2]{%
792     \csgdef{\glstr@preloctag@##1}{##2}%
793   }%
794   \renewcommand*{\@glstr@doloctag}{%
795     \ifcsundef{\glstr@preloctag\glscurrententrylabel}%
796     {%
797       \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
798         Rerun required}%
799     }%
800     {%
801       \csuse{\glstr@preloctag\glscurrententrylabel}%
802     }%
803   }%
804 }
805 \@onlypreamble\GlsXtrEnablePreLocationTag

```

`\glstrpreloctag`

```

806 \newcommand*{\@@glsxtrpreloctag}{%
807   \let\@glsxtr@org@delimN\delimN
808   \let\@glsxtr@org@delimR\delimR
809   \let\@glsxtr@org@glsignore\glsignore
  \gdef is required as the delimiters may occur inside a scope.
810   \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
811   \renewcommand*{\delimN}{%
812     \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
813     \@glsxtr@org@delimN}%
814   \renewcommand*{\delimR}{%
815     \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
816     \@glsxtr@org@delimR}%
817   \renewcommand*{\glsignore}[1]{%
818     \gdef\@glsxtr@thisloctag{\relax}%
819     \@glsxtr@org@glsignore{##1}}%
820   \@glsxtr@doloctag
821 }

```

glsxtrpreloctag

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

@glsxtr@pagetag

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

glsxtr@pagetag

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

lsxtrpostloctag

```

825 \newcommand*{\@@glsxtrpostloctag}{%
826   \let\delimN\@glsxtr@org@delimN
827   \let\delimR\@glsxtr@org@delimR
828   \let\glsignore\@glsxtr@org@glsignore
829   \protected@write\@auxout{}%
830   {\string\@glsxtr@savepreloctag{\glscurrententrylabel}\@glsxtr@thisloctag}}%
831 }

```

lsxtrpostloctag

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

lsxtr@preloctag

```

833 \newcommand*{\@glsxtr@savepreloctag}[2]{%
834   \protected@write\@auxout{}%
835   \string\providecommand\string\@glsxtr@savepreloctag[2]{}}

```

glsxtr@doloctag

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

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

```

837 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
838 \XKV@plfalse
839 \XKV@sttrue
840 \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
841 {%
842 \csname glsnonumberlist\XKV@resa\endcsname
843 \ifglsnonumberlist
844 \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
845 \else
846 \def\glossaryentrynumbers##1{%
847 \@glstrpreloctag
848 \GlsXtrFormatLocationList{##1}%
849 \@glstrpostloctag
850 \gls@save@numberlist{##1}}%
851 \fi
852 }%
853 }

```

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.

```

854 \renewcommand*{\glsentryfmt}{%
855 \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}}%
856 \glsifregular{\glslabel}%
857 {\glstrregularfont{\glsgenentryfmt}}%
858 {%
859 \ifglshasshort{\glslabel}%
860 {\glstrgenabbrvfmt}%
861 {\glstrregularfont{\glsgenentryfmt}}%
862 }%
863 }

```

sxtrregularfont Font used for regular entries.

```

864 \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.

```
865 \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 enter exists.

```
866 \@glstr@record{#2}{#3}%
```

```
867 \glsoifexists{#3}%
```

```
868 {%
```

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).

```
869 \let\glstrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
```

```
870 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
871 \def\glscustomtext{#4}%
```

```
872 \@glstr@field@linkdefs
```

```
873 #1%
```

```
874 \@gls@link[#2]{#3}{#4}%
```

```
875 \let\ifKV@glslink@hyper\glstrorg@ifKV@glslink@hyper
```

```
876 }%
```

```
877 \glspostlinkhook
```

```
878 }
```

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

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

```
879 \let\@glstr@org@gls@\@gls@
```

```
880 \def\@gls@#1#2{%
```

```
881 \@glstr@record{#1}{#2}%
```

```
882 \@glstr@org@gls@{#1}{#2}%
```

```
883 }%
```

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

```
884 \let\@glstr@org@glspl@\@glspl@
```

```
885 \def\@glspl@#1#2{%
```

```
886 \@glstr@record{#1}{#2}%
```

```
887 \@glstr@org@glspl@{#1}{#2}%
```

```
888 }%
```

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

```
889 \let\@glstr@org@Gls@\@Gls@
```

```
890 \def\@Gls@#1#2{%
```

```
891 \@glstr@record{#1}{#2}%
```

```
892 \@glstr@org@Gls@{#1}{#2}%
```

```
893 }%
```

\@Glspl@ Save the original definition and redefine.

```
894 \let\@glxtr@org@Glspl@\@Glspl@
895 \def\@Glspl@#1#2{%
896   \@glxtr@record{#1}{#2}%
897   \@glxtr@org@Glspl@{#1}{#2}%
898 }%
```

\@GLS@ Save the original definition and redefine.

```
899 \let\@glxtr@org@GLS@\@GLS@
900 \def\@GLS@#1#2{%
901   \@glxtr@record{#1}{#2}%
902   \@glxtr@org@GLS@{#1}{#2}%
903 }%
```

\@GLSpl@ Save the original definition and redefine.

```
904 \let\@glxtr@org@GLSpl@\@GLSpl@
905 \def\@GLSpl@#1#2{%
906   \@glxtr@record{#1}{#2}%
907   \@glxtr@org@GLSpl@{#1}{#2}%
908 }%
```

\@glsdisp1 Save the original definition and redefine.

```
909 \let\@glxtr@org@glsdisp\@glsdisp
910 \renewcommand*{\@glsdisp}[3] [] {%
911   \@glxtr@record{#1}{#2}%
912   \@glxtr@org@glsdisp[#1]{#2}{#3}%
913 }
```

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

```
914 \renewcommand*{\@gls@@link}[3] [] {%
915   \@glxtr@record{#1}{#2}%
916   \glsdoifexistsordo{#2}%
917   {%
918     \let\do@gls@link@checkfirsthyper\relax
919     \@gls@link[#1]{#2}{#3}%
920   }%
921   {%
922     \glstextformat{#3}%
923   }%
924   \glspostlinkhook
925 }
```

\glsadd Redefine to include \@glxtr@record

```
926 \renewrobustcmd*{\glsadd}[2] [] {%
927   \@gls@adjustmode
928   \@glxtr@record{#1}{#2}%
929   \glsdoifexists{#2}%
930   {%
```

```

931 \def\@glsnumberformat{glsnumberformat}%
932 \edef\@gls@counter{\csname glo@\glsdetoklabel{#2}@counter\endcsname}%
933 \setkeys{glossadd}{#1}%
934 \@gls@saveentrycounter
935 \@@do@wrglossary{#2}%
936 }%
937 }

```

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

```

938 \newcommand*{\@glsxtr@field@linkdefs}{%
939 \let\glsxtrifwasfirstuse\@secondoftwo
940 \let\glsifplural\@secondoftwo
941 \let\glsupcase\@firstofthree
942 \let\glsinsert\@empty
943 }

```

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

`assignfieldfont`

```

944 \newcommand*{\glsxtrassignfieldfont}[1]{%
945 \ifglstryexists{#1}%
946 {%
947 \ifglshasshort{#1}%
948 {%
949 \glssetabbrvfmt{\glscategory{#1}}%
950 \glsifregular{#1}%
951 {\let\@gls@field@font\glsxtrregularfont}%
952 {\let\@gls@field@font\@firstofone}%
953 }%
954 {%
955 \glsifnotregular{#1}%
956 {\let\@gls@field@font\@firstofone}%
957 {\let\@gls@field@font\glsxtrregularfont}%
958 }%
959 }%
960 {%
961 \let\@gls@field@font\@gobble
962 }%
963 }

```

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

```

964 \def\@glstext@#1#2[#3]{%
965 \glsxtrassignfieldfont{#2}%
966 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
967 }

```

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

```

968 \def\@GLStext@#1#2[#3]{%

```

```

969 \glstrassignfieldfont{#2}%
970 \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
971 {\@gls@field@font{\Glsaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
972 }

```

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

```

973 \def\@Glstext@#1#2[#3]{%
974 \glstrassignfieldfont{#2}%
975 \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
976 {\@gls@field@font{\Glsaccesstext{#2}#3}}%
977 }

```

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

`checknohyperfirst`

```

978 \newcommand*\glstrchecknohyperfirst[1]{%
979 \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
980 }

```

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

```

981 \def\@glsfirst@#1#2[#3]{%
982 \glstrassignfieldfont{#2}%
983 \glstrchecknohyperfirst{#2}%
984 [\let\glstrifwasfirstuse\@firstoftwo
985 \glstrchecknohyperfirst{#2}%
986 ]{#1}{#2}%
987 {\@gls@field@font{\Glsaccessfirst{#2}#3}}%
988 }

```

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

```

989 \def\@Glsfirst@#1#2[#3]{%
990 \glstrassignfieldfont{#2}%
991 \glstrchecknohyperfirst{#2}%
992 [\let\glstrifwasfirstuse\@firstoftwo
993 \let\glscapscase\@secondofthree
994 \glstrchecknohyperfirst{#2}%
995 ]%
996 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
997 }

```

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

```

998 \def\@GLSfirst@#1#2[#3]{%
999 \glstrassignfieldfont{#2}%

```

Ensure that \GLSfirst honours the nohyperfirst attribute.

```
1000 \@gls@field@link
1001 [\let\glstrifwasfirstuse\@firstoftwo
1002 \let\glscapscase\@thirdofthree
1003 \glstrchecknohyperfirst{#2}%
1004 ]%
1005 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1006 }
```

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

```
1007 \def\@glsplural@#1#2[#3]{%
1008 \glstrassignfieldfont{#2}%
1009 \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1010 {\@gls@field@font{\glsaccessplural{#2}#3}}%
1011 }
```

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

```
1012 \def\@Glsplural@#1#2[#3]{%
1013 \glstrassignfieldfont{#2}%
1014 \@gls@field@link
1015 [\let\glsifplural\@firstoftwo
1016 \let\glscapscase\@secondofthree
1017 ]%
1018 {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1019 }
```

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

```
1020 \def\@GLSplural@#1#2[#3]{%
1021 \glstrassignfieldfont{#2}%
1022 \@gls@field@link
1023 [\let\glsifplural\@firstoftwo
1024 \let\glscapscase\@thirdofthree
1025 ]%
1026 {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1027 }
```

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

```
1028 \def\glsfirstplural@#1#2[#3]{%
1029 \glstrassignfieldfont{#2}%
1030 \@gls@field@link
1031 [\let\glstrifwasfirstuse\@firstoftwo
1032 \let\glsifplural\@firstoftwo
1033 \glstrchecknohyperfirst{#2}%
1034 ]%
1035 {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1036 }
```

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

```
1037 \def\@Glsfirstplural@#1#2[#3]{%
1038   \glstrassignfieldfont{#2}%
    Ensure that \glfirstplural honours the nohyperfirst attribute.
1039   \@gls@field@link
1040   [\let\glstrifwasfirstuse\@firstoftwo
1041    \let\glstifplural\@firstoftwo
1042    \let\glscapscase\@secondofthree
1043    \glstrchecknohyperfirst{#2}%
1044   ]%
1045   {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1046 }
```

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

```
1047 \def\@GLSfirstplural@#1#2[#3]{%
1048   \glstrassignfieldfont{#2}%
    Ensure that \glfirstplural honours the nohyperfirst attribute.
1049   \@gls@field@link
1050   [\let\glstrifwasfirstuse\@firstoftwo
1051    \let\glstifplural\@firstoftwo
1052    \let\glscapscase\@thirdofthree
1053    \glstrchecknohyperfirst{#2}%
1054   ]%
1055   {#1}{#2}%
1056   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
1057 }
```

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

```
1058 \def\@glsname@#1#2[#3]{%
1059   \glstrassignfieldfont{#2}%
1060   \@gls@field@link{#1}{#2}{\@gls@field@font{\Glsaccessname{#2}#3}}%
1061 }
```

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

```
1062 \def\@Glsname@#1#2[#3]{%
1063   \glstrassignfieldfont{#2}%
1064   \@gls@field@link
1065   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1066   {\@gls@field@font{\Glsaccessname{#2}#3}}%
1067 }
```

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

```
1068 \def\@GLSname@#1#2[#3]{%
1069   \glstrassignfieldfont{#2}%
1070   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1071   {#1}{#2}%
1072   {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
1073 }
```

```

\@Glsdesc@
1074 \def\@Glsdesc@#1#2[#3]{%
1075   \glstrassignfieldfont{#2}%
1076   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
1077 }

\@GLSdesc@   First letter uppercase version.
1078 \def\@GLSdesc@#1#2[#3]{%
1079   \glstrassignfieldfont{#2}%
1080   \@gls@field@link
1081   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1082   {\@gls@field@font{\GLSaccessdesc{#2}#3}}%
1083 }

\@GLSdesc@   All uppercase version.
1084 \def\@GLSdesc@#1#2[#3]{%
1085   \glstrassignfieldfont{#2}%
1086   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1087   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
1088 }

@Glsdescplural@   No case-changing version.
1089 \def\@Glsdescplural@#1#2[#3]{%
1090   \glstrassignfieldfont{#2}%
1091   \@gls@field@link
1092   [\let\glscapscase\@secondoftwo
1093    \let\glsifplural\@firstoftwo
1094   ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
1095 }

@Glsdescplural@   First letter uppercase version.
1096 \def\@Glsdescplural@#1#2[#3]{%
1097   \glstrassignfieldfont{#2}%
1098   \@gls@field@link
1099   [\let\glscapscase\@secondoftwo
1100    \let\glsifplural\@firstoftwo
1101   ]{#1}{#2}{\@gls@field@font{\GLSaccessdescplural{#2}#3}}%
1102 }

@GLSdescplural@   All uppercase version.
1103 \def\@GLSdesc@#1#2[#3]{%
1104   \glstrassignfieldfont{#2}%
1105   \@gls@field@link
1106   [\let\glscapscase\@thirdoftwo
1107    \let\glsifplural\@firstoftwo
1108   ]%
1109   {#1}{#2}%
1110   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
1111 }

```

```

\@glssymbol@
1112 \def\@glssymbol@#1#2[#3]{%
1113   \glstrassignfieldfont{#2}%
1114   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
1115 }

\@Glsymbol@ First letter uppercase version.
1116 \def\@Glsymbol@#1#2[#3]{%
1117   \glstrassignfieldfont{#2}%
1118   \@gls@field@link
1119   [\let\glscapscase\@secondoftwo]%
1120   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
1121 }

\@GLSymbol@ All uppercase version.
1122 \def\@GLSymbol@#1#2[#3]{%
1123   \glstrassignfieldfont{#2}%
1124   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1125   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
1126 }

lssymbolplural@ No case-changing version.
1127 \def\@lssymbolplural@#1#2[#3]{%
1128   \glstrassignfieldfont{#2}%
1129   \@gls@field@link
1130   [\let\glscapscase\@secondoftwo
1131   \let\glsifplural\@firstoftwo
1132   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
1133 }

lssymbolplural@ First letter uppercase version.
1134 \def\@lssymbolplural@#1#2[#3]{%
1135   \glstrassignfieldfont{#2}%
1136   \@gls@field@link
1137   [\let\glscapscase\@secondoftwo
1138   \let\glsifplural\@firstoftwo
1139   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
1140 }

LSsymbolplural@ All uppercase version.
1141 \def\@LSsymbol@#1#2[#3]{%
1142   \glstrassignfieldfont{#2}%
1143   \@gls@field@link
1144   [\let\glscapscase\@thirdoftwo
1145   \let\glsifplural\@firstoftwo
1146   ]%
1147   {#1}{#2}%
1148   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
1149 }

```

\@Glsuseri@ First letter uppercase version.

```
1150 \def\@Glsuseri@#1#2[#3]{%
1151   \glstrassignfieldfont{#2}%
1152   \@gls@field@link
1153   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1154   {\@gls@field@font{\Glsentryuseri{#2}#3}}%
1155 }
```

\@GLSuseri@ All uppercase version.

```
1156 \def\@GLSuseri@#1#2[#3]{%
1157   \glstrassignfieldfont{#2}%
1158   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1159   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseri{#2}#3}}}%
1160 }
```

\@Glsuserii@ First letter uppercase version.

```
1161 \def\@Glsuserii@#1#2[#3]{%
1162   \glstrassignfieldfont{#2}%
1163   \@gls@field@link
1164   [\let\glscapscase\@secondoftwo]%
1165   {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
1166 }
```

\@GLSuserii@ All uppercase version.

```
1167 \def\@GLSuserii@#1#2[#3]{%
1168   \glstrassignfieldfont{#2}%
1169   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1170   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
1171 }
```

\@Glsuseriii@ First letter uppercase version.

```
1172 \def\@Glsuseriii@#1#2[#3]{%
1173   \glstrassignfieldfont{#2}%
1174   \@gls@field@link
1175   [\let\glscapscase\@secondoftwo]%
1176   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
1177 }
```

\@GLSuseriii@ All uppercase version.

```
1178 \def\@GLSuseriii@#1#2[#3]{%
1179   \glstrassignfieldfont{#2}%
1180   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1181   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
1182 }
```

\@Glsuseriv@ First letter uppercase version.

```
1183 \def\@Glsuseriv@#1#2[#3]{%
1184   \glstrassignfieldfont{#2}%
1185 }
```

```

1185 \@gls@field@link
1186 [\let\glscapscase\@secondoftwo]%
1187 {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
1188 }

```

\@GLSuseriv@ All uppercase version.

```

1189 \def\@GLSuseriv@#1#2[#3]{%
1190 \glstrassignfieldfont{#2}%
1191 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1192 {#1}{#2}%
1193 {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
1194 }

```

\@Glsuserv@ First letter uppercase version.

```

1195 \def\@Glsuserv@#1#2[#3]{%
1196 \glstrassignfieldfont{#2}%
1197 \@gls@field@link
1198 [\let\glscapscase\@secondoftwo]%
1199 {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
1200 }

```

\@GLSuserv@ All uppercase version.

```

1201 \def\@GLSuserv@#1#2[#3]{%
1202 \glstrassignfieldfont{#2}%
1203 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1204 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
1205 }

```

\@Glsuservi@ First letter uppercase version.

```

1206 \def\@Glsuservi@#1#2[#3]{%
1207 \glstrassignfieldfont{#2}%
1208 \@gls@field@link
1209 [\let\glscapscase\@secondoftwo]%
1210 {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}%
1211 }

```

\@GLSuservi@ All uppercase version.

```

1212 \def\@GLSuservi@#1#2[#3]{%
1213 \glstrassignfieldfont{#2}%
1214 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1215 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
1216 }

```

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

\@acrshort No case change.

```

1217 \def\@acrshort#1#2[#3]{%

```

```

1218 \glsdoifexists{#2}%
1219 {%
1220   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1221   \let\glxtrifwasfirstuse\@secondoftwo
1222   \let\gl@ifplural\@secondoftwo
1223   \let\glscapscase\@firstofthree
1224   \let\glinsert\@empty
1225   \def\glscustomtext{%
1226     \acronymfont{\glaccessshort{#2}}#3%
1227   }%
1228   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1229 }%
1230 \glspostlinkhook
1231 }

```

\@Acrshort First letter uppercase.

```

1232 \def\@Acrshort#1#2[#3]{%
1233   \glsdoifexists{#2}%
1234   {%
1235     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1236     \let\glxtrifwasfirstuse\@secondoftwo
1237     \let\gl@ifplural\@secondoftwo
1238     \let\glscapscase\@secondofthree
1239     \let\glinsert\@empty
1240     \def\glscustomtext{%
1241       \acronymfont{\Glsaccessshort{#2}}#3%
1242     }%
1243     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1244   }%
1245   \glspostlinkhook
1246 }

```

\@ACRshort All uppercase.

```

1247 \def\@ACRshort#1#2[#3]{%
1248   \glsdoifexists{#2}%
1249   {%
1250     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1251     \let\glxtrifwasfirstuse\@secondoftwo
1252     \let\gl@ifplural\@secondoftwo
1253     \let\glscapscase\@thirdofthree
1254     \let\glinsert\@empty
1255     \def\glscustomtext{%
1256       \mfirstucMakeUppercase{\acronymfont{\glaccessshort{#2}}#3}%
1257     }%
1258     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1259   }%
1260   \glspostlinkhook
1261 }

```

\@acrshortpl No case change.

```
1262 \def\@acrshortpl#1#2[#3]{%
1263   \glsdoifexists{#2}%
1264   {%
1265     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1266     \let\glxtrifwasfirstuse\@secondoftwo
1267     \let\gl@sifplural\@firstoftwo
1268     \let\glscapscase\@firstofthree
1269     \let\gl$insert\@empty
1270     \def\glscustomtext{%
1271       \acronymfont{\gl@saccesssshortpl{#2}}#3%
1272     }%
1273     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1274   }%
1275   \glspostlinkhook
1276 }
```

\@Acrshortpl First letter uppercase.

```
1277 \def\@Acrshortpl#1#2[#3]{%
1278   \glsdoifexists{#2}%
1279   {%
1280     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1281     \let\glxtrifwasfirstuse\@secondoftwo
1282     \let\gl@sifplural\@firstoftwo
1283     \let\glscapscase\@secondofthree
1284     \let\gl$insert\@empty
1285     \def\glscustomtext{%
1286       \acronymfont{\Glsaccesssshortpl{#2}}#3%
1287     }%
1288     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1289   }%
1290   \glspostlinkhook
1291 }
```

\@ACRshortpl All uppercase.

```
1292 \def\@ACRshortpl#1#2[#3]{%
1293   \glsdoifexists{#2}%
1294   {%
1295     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1296     \let\glxtrifwasfirstuse\@secondoftwo
1297     \let\gl@sifplural\@firstoftwo
1298     \let\glscapscase\@thirdofthree
1299     \let\gl$insert\@empty
1300     \def\glscustomtext{%
1301       \mfirstucMakeUppercase{\acronymfont{\gl@saccesssshortpl{#2}}#3}%
1302     }%
1303     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1304   }%
1305   \glspostlinkhook
```

1306 }

\@acrlong No case change.

```
1307 \def\@acrlong#1#2[#3]{%
1308   \glsdoifexists{#2}%
1309   {%
1310     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1311     \let\glxtrifwasfirstuse\@secondoftwo
1312     \let\glsifplural\@secondoftwo
1313     \let\glscapscase\@firstofthree
1314     \let\glsinsert\@empty
1315     \def\glscustomtext{%
1316       \acronymfont{\glsaccesslong{#2}}#3%
1317     }%
1318     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1319   }%
1320   \glspostlinkhook
1321 }
```

\@Acrlong First letter uppercase.

```
1322 \def\@Acrlong#1#2[#3]{%
1323   \glsdoifexists{#2}%
1324   {%
1325     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1326     \let\glxtrifwasfirstuse\@secondoftwo
1327     \let\glsifplural\@secondoftwo
1328     \let\glscapscase\@secondofthree
1329     \let\glsinsert\@empty
1330     \def\glscustomtext{%
1331       \acronymfont{\Glsaccesslong{#2}}#3%
1332     }%
1333     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1334   }%
1335   \glspostlinkhook
1336 }
```

\@ACRlong All uppercase.

```
1337 \def\@ACRlong#1#2[#3]{%
1338   \glsdoifexists{#2}%
1339   {%
1340     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1341     \let\glxtrifwasfirstuse\@secondoftwo
1342     \let\glsifplural\@secondoftwo
1343     \let\glscapscase\@thirdofthree
1344     \let\glsinsert\@empty
1345     \def\glscustomtext{%
1346       \mfirstucMakeUppercase{\acronymfont{\glsaccesslong{#2}}#3}%
1347     }%
1348     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1349   }
```

```

1349 }%
1350 \glspostlinkhook
1351 }

```

\@acrlongpl No case change.

```

1352 \def\@acrlongpl#1#2[#3]{%
1353   \glsdoidexists{#2}%
1354   {%
1355     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1356     \let\glxtrifwasfirstuse\@secondoftwo
1357     \let\gl@sifplural\@firstoftwo
1358     \let\glscapscase\@firstofthree
1359     \let\gl$insert\@empty
1360     \def\glscustomtext{%
1361       \acronymfont{\gl@saccesslongpl{#2}}#3%
1362     }%
1363     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1364   }%
1365   \glspostlinkhook
1366 }

```

\@Acrlongpl First letter uppercase.

```

1367 \def\@Acrlongpl#1#2[#3]{%
1368   \glsdoidexists{#2}%
1369   {%
1370     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1371     \let\glxtrifwasfirstuse\@secondoftwo
1372     \let\gl@sifplural\@firstoftwo
1373     \let\glscapscase\@secondofthree
1374     \let\gl$insert\@empty
1375     \def\glscustomtext{%
1376       \acronymfont{\Glsaccesslongpl{#2}}#3%
1377     }%
1378     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1379   }%
1380   \glspostlinkhook
1381 }

```

\@ACRlongpl All uppercase.

```

1382 \def\@ACRlongpl#1#2[#3]{%
1383   \glsdoidexists{#2}%
1384   {%
1385     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1386     \let\glxtrifwasfirstuse\@secondoftwo
1387     \let\gl@sifplural\@firstoftwo
1388     \let\glscapscase\@thirdofthree
1389     \let\gl$insert\@empty
1390     \def\glscustomtext{%
1391       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslongpl{#2}}#3}%

```

```

1392 }%
1393 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1394 }%
1395 \glspostlinkhook
1396 }

```

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

\@glsaddkey

```

1397 \renewcommand*{\@glsaddkey}[7]{%
1398 \key@ifundefined{glossentry}{#1}%
1399 {%
1400 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1401 \appto\@gls@keymap{,{#1}{#1}}%
1402 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
1403 \appto\@newglossaryentryposthook{%
1404 \letcs{\@glo@tmp}{@glo@#1}%
1405 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1406 }%
1407 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1408 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

1409 \ifcsdef{@gls@user@#1@}%
1410 {%
1411 \PackageError{glossaries}%
1412 {Can't define '\string#5' as helper command
1413 '\expandafter\string\csname @gls@user@#1@\endcsname' already
1414 exists}%
1415 }%
1416 }%
1417 {%
1418 \expandafter\newcommand\expandafter*\expandafter
1419 {\csname @gls@user@#1\endcsname}[2][ ]{%
1420 \new@ifnextchar[%
1421 {\csuse{@gls@user@#1@}{##1}{##2}}%
1422 {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
1423 \csdef{@gls@user@#1@}##1##2[##3]{%
1424 \@gls@field@link{##1}{##2}{#3{##2}##3}%
1425 }%
1426 \newrobustcmd*{#5}{%
1427 \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
1428 }%

```

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

```

1429 \ifcsdef{@Gls@user@#1@}%
1430 {%
1431 \PackageError{glossaries}%
1432 {Can't define '\string#6' as helper command

```

```

1433         '\expandafter\string\csname @Gls@user@#1@endcsname' already
1434         exists}%
1435     {}%
1436 }%
1437 {%
1438     \expandafter\newcommand\expandafter*\expandafter
1439     {\csname @Gls@user@#1@endcsname}[2][\%
1440         \new@ifnextchar[%
1441             {\csuse{@Gls@user@#1@}{##1}{##2}}}%
1442             {\csuse{@Gls@user@#1@}{##1}{##2}[]}}}%
1443     \csdef{@Gls@user@#1@}##1##2[##3]{%
1444         \@gls@field@link[\let\glscapscase\@secondofthree]%
1445         {##1}{##2}{#4{##2}##3}%
1446     }%
1447     \newrobustcmd*{#6}{%
1448         \expandafter\@gls@hyp@opt\csname @Gls@user@#1@endcsname}%
1449     }%

```

Finally the all caps version (modified):

```

1450     \ifcsdef{@GLS@user@#1@}%
1451     {%
1452         \PackageError{glossaries}%
1453         {Can't define '\string#7' as helper command
1454         '\expandafter\string\csname @GLS@user@#1@endcsname' already
1455         exists}%
1456     {}%
1457 }%
1458 {%
1459     \expandafter\newcommand\expandafter*\expandafter
1460     {\csname @GLS@user@#1@endcsname}[2][\%
1461         \new@ifnextchar[%
1462             {\csuse{@GLS@user@#1@}{##1}{##2}}}%
1463             {\csuse{@GLS@user@#1@}{##1}{##2}[]}}}%
1464     \csdef{@GLS@user@#1@}##1##2[##3]{%
1465         \@gls@field@link[\let\glscapscase\@thirdofthree]%
1466         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}}%
1467     }%
1468     \newrobustcmd*{#7}{%
1469         \expandafter\@gls@hyp@opt\csname @GLS@user@#1@endcsname}%
1470     }%
1471 }%
1472 {%
1473     \PackageError{glossaries-extra}{Key '#1' already exists}{}%
1474 }%
1475 }

```

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

```

1476 \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.

```
1477 \let\@glstr@org@checkfirsthyper\@gls@link@checkfirsthyper
1478 \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.

```
1479 \ifglsused{\glslabel}%
1480 {\let\glstr@ifwasfirstuse\@secondoftwo}
1481 {\let\glstr@ifwasfirstuse\@firstoftwo}%
```

Store the category label for convenience.

```
1482 \edef\glscategorylabel{\glscategory{\glslabel}}%
1483 \ifglsused{\glslabel}%
1484 {%
1485   \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
1486   {\KV@glslink@hyperfalse}{}%
1487 }%
1488 {%
1489   \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
1490   {\KV@glslink@hyperfalse}{}%
1491 }%
1492 \glslinkcheckfirsthyperhook
1493 }
```

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.

```
1494 \ifdef\do@glstdisablehyperinlist
1495 {%
1496   \let\@glstr@do@glstdisablehyperinlist\do@glstdisablehyperinlist
1497   \renewcommand*{\do@glstdisablehyperinlist}{%
1498     \@glstr@do@glstdisablehyperinlist
1499     \glsifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}%
1500   }
1501 }
1502 }
```

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

```
1503 \define@boolkey{glslink}{noindex}[true]{}
1504 \KV@glslink@noindexfalse
```

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

lt@glslink@opts

```
1505 \ifdef\@gls@setdefault@glslink@opts
1506 {
1507   \renewcommand*{\@gls@setdefault@glslink@opts}{%
1508     \KV@glslink@noindexfalse
```

```

1509 }
1510 }
1511 {
    Not defined so prepend it to \do@glssdisablehyperinlist to achieve the same effect.
1512 \newcommand*{\@gls@setdefault@glslink@opts}{%
1513     \KV@glslink@noindexfalse
1514 }
1515 \preto\do@glssdisablehyperinlist{\@gls@setdefault@glslink@opts}
1516 }

```

DefaultGlsOpts Set the default options for \glslink etc.

```

1517 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
1518     \renewcommand*{\@gls@setdefault@glslink@opts}{\setkeys{glslink}{#1}}%
1519 }

```

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

```

1520 \newcommand*{\glstrifindexing}[2]{%
1521     \ifKV@glslink@noindex #2\else #1\fi
1522 }

```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```

1523 \renewcommand*{\glswriteentry}[2]{%
1524     \glstrifindexing
1525     {%
1526         \ifglssindexonlyfirst
1527             \ifglssused{#1}
1528                 {\glstrdoautoindexname{#1}{dualindex}}%
1529                 {#2}%
1530         \else
1531             \glssifattribute{#1}{indexonlyfirst}{true}%
1532             {\ifglssused{#1}
1533                 {\glstrdoautoindexname{#1}{dualindex}}%
1534                 {#2}}%
1535             {#2}%
1536         \fi
1537     }%
1538     {}%
1539 }

```

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

```

1540 \appto\@do@@wrglossary{\@glstrdo@@wrindex
1541     \glstrdowrglossaryhook{\@gls@label}%
1542 }

```

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

Similarly for the “noidx” version:

s@noidxglossary

```
1543 \appto\gls@noidxglossary{\@glstr@do@@wrindex
1544 \glstrdowrglossaryhook{\@gls@label}}%
1545 }
```

xtr@do@@wrindex

```
1546 \newcommand*{\@glstr@do@@wrindex}{%
1547 \glstrdoautoindexname{\@gls@label}{dualindex}}%
1548 }
```

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.)

```
1549 \newcommand*{\glstrdowrglossaryhook}[1]{}
```

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

```
1550 \newcommand*{\@gls@alt@hyp@opt}[1]{%
1551 \let\glslinkvar\@firstofthree
1552 \let\@gls@hyp@opt@cs#1\relax
1553 \@ifstar{\s@gls@hyp@opt}%
1554 {\@ifnextchar+%
1555 {\@firstoftwo{\p@gls@hyp@opt}}%
1556 {%
1557 \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
1558 {\@firstoftwo{\@alt@gls@hyp@opt}}%
1559 {\#1}%
1560 }%
1561 }%
1562 }
```

alt@gls@hyp@opt User version

```
1563 \newcommand*{\@alt@gls@hyp@opt}[1][ ]{%
1564 \let\glslinkvar\@firstofthree
1565 \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}
```

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

```
1566 \newcommand*{\@gls@alt@hyp@opt@char}{}
```

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

```
1567 \newcommand*{\@gls@alt@hyp@opt@keys}{}
```

rSetAltModifier

```
1568 \newcommand*{\GlsXtrSetAltModifier}[2]{%
1569 \let\@gls@hyp@opt\@gls@alt@hyp@opt
1570 \def\@gls@alt@hyp@opt@char{\#1}%
1571 \def\@gls@alt@hyp@opt@keys{\#2}%
1572 }
```

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

```

1573 \renewcommand*{\glsdohyperlink}[2]{%
1574   \glsattribute{\glslabel}{targeturl}%
1575   {%
1576     \glsattribute{\glslabel}{targetname}%
1577     {%
1578       \glsattribute{\glslabel}{targetcategory}%
1579       {%
1580         \hyperref{\glsattribute{\glslabel}{targeturl}}%
1581           {\glsattribute{\glslabel}{targetcategory}}%
1582           {\glsattribute{\glslabel}{targetname}}%
1583           {\glxtrprotectlinks#2}}%
1584       }%
1585     }%
1586     \hyperref{\glsattribute{\glslabel}{targeturl}}%
1587       {%
1588         {\glsattribute{\glslabel}{targetname}}%
1589         {\glxtrprotectlinks#2}}%
1590     }%
1591   }%
1592   {%
1593     \href{\glsattribute{\glslabel}{targeturl}}%
1594       {\glxtrprotectlinks#2}}%
1595   }%
1596 }%
1597 {%
1598   \hyperlink{#1}{\glxtrprotectlinks#2}}%
1599 }%
1600 }

```

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

```

1601 \ifundef\glsdohyperlink
1602 {%
1603   \renewcommand{\glsdisablehyper}{%
1604     \KV@glslink@hyperfalse
1605     \let\@glslink\glsdohyperlink
1606     \let\@glstarget\@secondoftwo
1607   }
1608 }
1609 {}

```

`\glsdohyperlink` 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.

```
1610 \def\glsdonohyperlink#1#2{{\glstrprotectlinks #2}}
```

Reset \@glslink with patched versions:

```
1611 \ifcsundef{hyperlink}%
1612 {%
1613   \let\@glslink\glsdonohyperlink
1614 }%
1615 {%
1616   \let\@glslink\glsdohyperlink
1617 }
```

glstrprotectlinks Make \gls (and variants) behave like the corresponding \gls{text} (and variants) with hyperlinking and indexing off.

```
1618 \newcommand*{\glstrprotectlinks}{%
1619   \KV@glslink@hyperfalse
1620   \KV@glslink@noindextrue
1621   \let\@gls@\@glstr@p@text@
1622   \let\@Gls@\@Glsxtr@p@text@
1623   \let\@GLS@\@GLSxtr@p@text@
1624   \let\@glspl@\@glstr@p@plural@
1625   \let\@Glspl@\@Glsxtr@p@plural@
1626   \let\@GLSpl@\@GLSxtr@p@plural@
1627   \let\@glstrshort@\@glstr@p@short@
1628   \let\@Glsxtrshort@\@Glsxtr@p@short@
1629   \let\@GLSxtrshort@\@GLSxtr@p@short@
1630   \let\@glstrlong@\@glstr@p@long@
1631   \let\@Glsxtrlong@\@Glsxtr@p@long@
1632   \let\@GLSxtrlong@\@GLSxtr@p@long@
1633   \let\@glstrshortpl@\@glstr@p@shortpl@
1634   \let\@Glsxtrshortpl@\@Glsxtr@p@shortpl@
1635   \let\@GLSxtrshortpl@\@GLSxtr@p@shortpl@
1636   \let\@glstrlongpl@\@glstr@p@longpl@
1637   \let\@Glsxtrlongpl@\@Glsxtr@p@longpl@
1638   \let\@GLSxtrlongpl@\@GLSxtr@p@longpl@
1639   \let\@acrshort@\@glstr@p@acrshort@
1640   \let\@Acrshort@\@Glsxtr@p@acrshort@
1641   \let\@ACRshort@\@GLSxtr@p@acrshort@
1642   \let\@acrshortpl@\@glstr@p@acrshortpl@
1643   \let\@Acrshortpl@\@Glsxtr@p@acrshortpl@
1644   \let\@ACRshortpl@\@GLSxtr@p@acrshortpl@
1645   \let\@acrlong@\@glstr@p@acrlong@
1646   \let\@Acrlong@\@Glsxtr@p@acrlong@
1647   \let\@ACRlong@\@GLSxtr@p@acrlong@
1648   \let\@acrlongpl@\@glstr@p@acrlongpl@
1649   \let\@Acrlongpl@\@Glsxtr@p@acrlongpl@
1650   \let\@ACRlongpl@\@GLSxtr@p@acrlongpl@
1651 }
```

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

```
@glxtr@p@text@
1652 \def\@glxtr@p@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}}

@Glsxtr@p@text@
1653 \def\@Glsxtr@p@text@#1#2[#3]{\@GLstext@{#1}{#2}[#3]}}

@GLSxtr@p@text@
1654 \def\@GLSxtr@p@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}}

lsxtr@p@plural@
1655 \def\@glxtr@p@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}}

lsxtr@p@plural@
1656 \def\@Glsxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}}

LSxtr@p@plural@
1657 \def\@GLSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}}

glxtr@p@short@
1658 \def\@glxtr@p@short@#1#2[#3]{%
1659 {%
1660   \glsssetabbrvfmt{\glscategory{#2}}%
1661   \glssabbrvfont{\glssentryshort{#2}}#3%
1662 }%
1663 }

Glsxtr@p@short@
1664 \def\@Glsxtr@p@short@#1#2[#3]{%
1665 {%
1666   \glsssetabbrvfmt{\glscategory{#2}}%
1667   \glssabbrvfont{\Glsentryshort{#2}}#3%
1668 }%
1669 }

GLSxtr@p@short@
1670 \def\@GLSxtr@p@short@#1#2[#3]{%
1671 {%
1672   \glsssetabbrvfmt{\glscategory{#2}}%
1673   \mfirstucMakeUppercase{\glssabbrvfont{\glssentryshort{#2}}#3}%
1674 }%
1675 }

sxtr@p@shortpl@
1676 \def\@glxtr@p@shortpl@#1#2[#3]{%
1677 {%
1678   \glsssetabbrvfmt{\glscategory{#2}}%
```

```

1679 \glsabbrvfont{\glsentryshortpl{#2}}#3%
1680 }%
1681 }

sctr@p@shortpl@
1682 \def\@Glsctr@p@shortpl@#1#2[#3]{%
1683 {%
1684 \glssetabbrvfmt{\glsentryshortpl{#2}}#3%
1685 \glsabbrvfont{\glsentryshortpl{#2}}#3%
1686 }%
1687 }

Sctr@p@shortpl@
1688 \def\@GLSctr@p@shortpl@#1#2[#3]{%
1689 {%
1690 \glssetabbrvfmt{\glsentryshortpl{#2}}#3%
1691 \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshortpl{#2}}#3}%
1692 }%
1693 }

@glscctr@p@long@
1694 \def\@glscctr@p@long@#1#2[#3]{\glsentrylong{#2}#3}

@Glsctr@p@long@
1695 \def\@Glsctr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}

@GLSctr@p@long@
1696 \def\@GLSctr@p@long@#1#2[#3]{%
1697 {\mfirstucMakeUppercase{\glsentrylong{#2}#3}}}

lsctr@p@longpl@
1698 \def\@glscctr@p@longpl@#1#2[#3]{\glsentrylongpl{#2}#3}

LSctr@p@longpl@
1699 \def\@Glsctr@p@longpl@#1#2[#3]{\Glsentrylongpl{#2}#3}

LScctr@p@longpl@
1700 \def\@GLScctr@p@longpl@#1#2[#3]{%
1701 {\mfirstucMakeUppercase{\glsentrylongpl{#2}#3}}}

xctr@p@acrshort@
1702 \def\@glscctr@p@acrshort@#1#2[#3]{\acronymfont{\glsentryshort{#2}}#3}

Xctr@p@acrshort@
1703 \def\@Glsctr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}

Xcctr@p@acrshort@
1704 \def\@GLScctr@p@acrshort@#1#2[#3]{%
1705 {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

```

```

r@p@acrshortpl@
1706 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
1707 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
1708 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
1709   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}}

sxtr@p@acrlong@
1710 \def\@glsxtr@p@acrlong@#1#2[#3]{\glsentrylong{#2}#3}}

sxtr@p@acrlong@
1711 \def\@Glsxtr@p@acrlong@#1#2[#3]{\Glsentrylong{#2}#3}}

Sxtr@p@acrlong@
1712 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
1713   {\mfirstucMakeUppercase{\glsentrylong{#2}#3}}}}

tr@p@acrlongpl@
1714 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\glsentrylongpl{#2}#3}}

tr@p@acrlongpl@
1715 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\Glsentrylongpl{#2}#3}}

tr@p@acrlongpl@
1716 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
1717   {\mfirstucMakeUppercase{\glsentrylongpl{#2}#3}}}}

Commands to minimise conflict.

\@glsxtrp@opt
1718 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

\glsxtrsetpopts Used in glossary to switch hyperlinks on for the \glsxtrp type of commands.
1719 \newcommand*{\glsxtrsetpopts}[1]{%
1720   \renewcommand*{\@glsxtrp@opt}{#1}%
1721 }

lossxtrsetpopts Used in glossary to switch hyperlinks on for the \glsxtrp type of commands.
1722 \newcommand*{\glossxtrsetpopts}{%
1723   \glsxtrsetpopts{noindex}%
1724 }

\@@glsxtrp
1725 \newrobustcmd*{\@@glsxtrp}[2]{%

```

Add scope.

```
1726 {%
1727   \let\glspostlinkhook\relax
1728   \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
1729 }%
1730 }
```

\@glsxtrp

```
1731 \newrobustcmd*{\@glsxtrp}[2]{%
1732   \ifcsdef{gls#1}%
1733   {%
1734     \@glsxtrp{gls#1}{#2}%
1735   }%
1736   {%
1737     \ifcsdef{glsxtr#1}%
1738     {%
1739       \@glsxtrp{glsxtr#1}{#2}%
1740     }%
1741     {%
1742       \PackageError{glossaries-extra}{‘#1’ not recognised by
1743         \string\glsxtrp}{}%
1744     }%
1745   }%
1746 }
```

\@Glsxtrp

```
1747 \newrobustcmd*{\@Glsxtrp}[2]{%
1748   \ifcsdef{Gls#1}%
1749   {%
1750     \@glsxtrp{Gls#1}{#2}%
1751   }%
1752   {%
1753     \ifcsdef{Glsxtr#1}%
1754     {%
1755       \@glsxtrp{Glsxtr#1}{#2}%
1756     }%
1757     {%
1758       \PackageError{glossaries-extra}{‘#1’ not recognised by
1759         \string\Glsxtrp}{}%
1760     }%
1761   }%
1762 }
```

\@GLSxtrp

```
1763 \newrobustcmd*{\@GLSxtrp}[2]{%
1764   \ifcsdef{GLS#1}%
1765   {%
1766     \@glsxtrp{GLS#1}{#2}%
1767   }%
```

```

1768 {%
1769   \ifcsdef{GLSxtr#1}%
1770   {%
1771     \@@glsxtrp{GLSxtr#1}{#2}%
1772   }%
1773   {%
1774     \PackageError{glossaries-extra}{‘#1’ not recognised by
1775       \string\GLSxtrp}{}%
1776   }%
1777 }%
1778 }

```

\glsxtr@entry@p

```

1779 \newrobustcmd*{\glsxtr@headentry@p}[2]{%
1780   \glsifattribute{#1}{headuc}{true}%
1781   {%
1782     \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
1783   }%
1784   {%
1785     \@gls@entry@field{#1}{#2}%
1786   }%
1787 }

```

\glsxtrp Not robust as it needs to expand somewhat.

```

1788 \ifdef\texorpdfstring
1789 {
1790   \newcommand{\glsxtrp}[2]{%
1791     \protect\NoCaseChange
1792     {%
1793       \protect\texorpdfstring
1794       {%
1795         \protect\glsxtrifinmark
1796         {%
1797           \ifcsdef{glsxtrhead#1}%
1798           {%
1799             {\protect\csuse{glsxtrhead#1}{#2}}%
1800           }%
1801           {%
1802             \glsxtr@headentry@p{#2}{#1}%
1803           }%
1804         }%
1805         {%
1806           \@glsxtrp{#1}{#2}%
1807         }%
1808       }%
1809       {%
1810         \protect\@gls@entry@field{#2}{#1}%
1811       }%
1812     }%

```

```

1813 }
1814 }
1815 {
1816   \newcommand{\glxtrp}[2]{%
1817     \protect\NoCaseChange
1818     {%
1819       \protect\glxtrifinmark
1820       {%
1821         \ifcsdef{glxtrhead#1}%
1822         {%
1823           {\protect\csuse{glxtrhead#1}}%
1824         }%
1825         {%
1826           \glxtr@headentry@p{#2}{#1}%
1827         }%
1828       }%
1829     }%
1830     \@glxtrp{#1}{#2}%
1831   }%
1832 }%
1833 }
1834 }

```

Provide short synonyms for the most common option.

`\glsp`

```
1835 \newcommand*{\glsp}{\glxtrp{short}}
```

`\glsp`

```
1836 \newcommand*{\glsp}{\glxtrp{text}}
```

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

```

1837 \ifdef\txorpdfstring
1838 {
1839   \newcommand{\Glsxtrp}[2]{%
1840     \protect\NoCaseChange
1841     {%
1842       \protect\txorpdfstring
1843       {%
1844         \protect\glxtrifinmark
1845         {%
1846           \ifcsdef{Glsxtrhead#1}%
1847           {%
1848             {\protect\csuse{Glsxtrhead#1}{#2}}%
1849           }%
1850           {%
1851             \protect\@Gls@entry@field{#2}{#1}%
1852           }%

```

```

1853     }%
1854     {%
1855         \@Glsxtrp{#1}{#2}%
1856     }%
1857 }%
1858 {%
1859     \protect\@gls@entry@field{#2}{#1}%
1860 }%
1861 }%
1862 }
1863 }
1864 {
1865     \newcommand{\Glsxtrp}[2]{%
1866         \protect\NoCaseChange
1867         {%
1868             \protect\glsxtrifinmark
1869             {%
1870                 \ifcsdef{Glsxtrhead#1}%
1871                 {%
1872                     {\protect\csuse{Glsxtrhead#1}}%
1873                 }%
1874                 {%
1875                     \protect\@Gls@entry@field{#2}{#1}%
1876                 }%
1877             }%
1878             {%
1879                 \@Glsxtrp{#1}{#2}%
1880             }%
1881         }%
1882     }
1883 }

```

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

```

1884 \ifdef\texorpdfstring
1885 {
1886     \newcommand{\GLSxtrp}[2]{%
1887         \protect\NoCaseChange
1888         {%
1889             \protect\texorpdfstring
1890             {%
1891                 \protect\glsxtrifinmark
1892                 {%
1893                     \ifcsdef{GLSxtr#1}%
1894                     {%
1895                         {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
1896                     }%
1897                     {%
1898                         \protect\mfirstucMakeUppercase
1899                     }%

```

```

1900         \protect\@gls@entry@field{#2}{#1}%
1901     }%
1902 }%
1903 }%
1904 {%
1905     \@GLSxtrp{#1}{#2}%
1906 }%
1907 }%
1908 {%
1909     \protect\@gls@entry@field{#2}{#1}%
1910 }%
1911 }%
1912 }
1913 }
1914 {
1915     \newcommand{\GLSxtrp}[2]{%
1916         \protect\NoCaseChange
1917         {%
1918             \protect\glsxtrifinmark
1919             {%
1920                 \ifcsdef{GLSxtr#1}%
1921                 {%
1922                     {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
1923                 }%
1924                 {%
1925                     \protect\mfirstucMakeUppercase
1926                     {%
1927                         \protect\@gls@entry@field{#2}{#1}%
1928                     }%
1929                 }%
1930             }%
1931             {%
1932                 \@GLSxtrp{#1}{#2}%
1933             }%
1934         }%
1935     }
1936 }

```

1.3.5 Entry Counting

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

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

```

\@glsunset  Global unset.
1937 \renewcommand*{\@glsunset}[1]{%
1938     \@glsunset{#1}%
1939     \glsxtrpostunset{#1}%

```

```

1940 }%

glxstrpostunset
1941 \newcommand*{\glxstrpostunset}[1]{}

\@glslocalunset Local unset.
1942 \renewcommand*{\@glslocalunset}[1]{%
1943 \@@glslocalunset{#1}%
1944 \glxstrpostlocalunset{#1}%
1945 }%

rpostlocalunset
1946 \newcommand*{\glxstrpostlocalunset}[1]{}

\@glsreset Global reset.
1947 \renewcommand*{\@glsreset}[1]{%
1948 \@@glsreset{#1}%
1949 \glxstrpostreset{#1}%
1950 }%

glxstrpostreset
1951 \newcommand*{\glxstrpostreset}[1]{}

\@glslocalreset Local reset.
1952 \renewcommand*{\@glslocalreset}[1]{%
1953 \@@glslocalreset{#1}%
1954 \glxstrpostlocalreset{#1}%
1955 }%

rpostlocalreset
1956 \newcommand*{\glxstrpostlocalreset}[1]{}

leEntryCounting The first argument is the list of categories and the second argument is the value of the en-
trycount attribute.
1957 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
Enable entry counting:
1958 \glsenableentrycount
Redefine \gls etc:
1959 \renewcommand*{\gls}{\cgl}%
1960 \renewcommand*{\Gls}{\cGls}%
1961 \renewcommand*{\glspl}{\cglspl}%
1962 \renewcommand*{\Glspl}{\cGlspl}%
1963 \renewcommand*{\GLS}{\cGLS}%
1964 \renewcommand*{\GLSpl}{\cGLSpl}%
Set the entrycount attribute:
1965 \@glxstr@setentrycountunsetattr{#1}{#2}%

```

In case this command is used again:

```

1966 \let\GlsXtrEnableEntryCounting\@glxtr@setentrycountunsetattr
1967 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
1968 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
1969 can't be used with \string\GlsXtrEnableEntryCounting}%
1970 {Use one or other but not both commands}}%
1971 }

```

ycountunsetattr

```

1972 \newcommand*{\@glxtr@setentrycountunsetattr}[2]{%
1973 \@for\@glxtr@cat:=#1\do
1974 {%
1975 \ifdefempty{\@glxtr@cat}{}%
1976 {%
1977 \glsssetcategoryattribute{\@glxtr@cat}{entrycount}{#2}%
1978 }%
1979 }%
1980 }

```

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

nableentrycount

```

1981 \renewcommand*{\glsenableentrycount}{%

```

Enable new fields:

```

1982 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%

```

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

```

1983 \renewcommand*{\gls@defdocnewglossaryentry}{%
1984 \renewcommand*\newglossaryentry[2]{%
1985 \PackageError{glossaries}{\string\newglossaryentry\space
1986 may only be used in the preamble when entry counting has
1987 been activated}{If you use \string\glsenableentrycount\space
1988 you must place all entry definitions in the preamble not in
1989 the document environment}%
1990 }%
1991 }%

```

New commands to access new fields:

```

1992 \newcommand*{\glsentrycurrcount}[1]{%
1993 \ifcsundef{glo@\glsdetoklabel{##1}@currcount}%
1994 {0}{\@gls@entry@field{##1}{currcount}}%
1995 }%
1996 \newcommand*{\glsentryprevcount}[1]{%
1997 \ifcsundef{glo@\glsdetoklabel{##1}@prevcount}%
1998 {0}{\@gls@entry@field{##1}{prevcount}}%
1999 }%

```

Adjust post unset and reset:

```

2000 \let\@glxtr@entrycount@org@unset\glxtrpostunset
2001 \renewcommand*{\glxtrpostunset}[1]{%

```

```

2002 \@glsxtr@entrycount@org@unset{##1}%
2003 \@gls@increment@currcount{##1}%
2004 }%
2005 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
2006 \renewcommand*{\glsxtrpostlocalunset}[1]{%
2007 \@glsxtr@entrycount@org@localunset{##1}%
2008 \@gls@local@increment@currcount{##1}%
2009 }%
2010 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
2011 \renewcommand*{\glsxtrpostreset}[1]{%
2012 \@glsxtr@entrycount@org@reset{##1}%
2013 \csgdef{glo@glsdetoklabel{##1}@currcount}{0}%
2014 }%
2015 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
2016 \renewcommand*{\glsxtrpostlocalreset}[1]{%
2017 \@glsxtr@entrycount@org@localreset{##1}%
2018 \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
2019 }%

```

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

```

2020 \let\@cgls@\@cgls@
2021 \let\@cglspl@\@cglspl@
2022 \let\@cGLS@\@cGLS@
2023 \let\@cGLspl@\@cGLspl@
2024 \let\@cGLS@\@cGLS@
2025 \let\@cGLSpl@\@cGLSpl@

```

The rest is as the original definition.

```

2026 \AtEndDocument{\@gls@write@entrycounts}%
2027 \renewcommand*{\@gls@entry@count}[2]{%
2028 \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
2029 }%
2030 \let\glsenableentrycount\relax
2031 \renewcommand*{\glsenableentryunitcount}{%
2032 \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space
2033 can't be used with \string\glsenableentrycount}%
2034 {Use one or other but not both commands}%
2035 }%
2036 }

```

`\@gls@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.

```

2037 \renewcommand*{\@gls@write@entrycounts}{%
2038 \immediate\write\@auxout
2039 {\string\providecommand*{\string\@gls@entry@count}[2]{}}%
2040 \count@=0\relax
2041 \forallglsentries{\@glsentry}{%
2042 \gls@hasattribute{\@glsentry}{entrycount}%
2043 }%

```

```

2044 \ifglused{\@glentry}%
2045 {%
2046 \immediate\write\@auxout
2047 {\string\@gl@entry@count{\@glentry}{\glentrycurrcount{\@glentry}}}%
2048 }%
2049 {}%
2050 \advance\count@ by \@ne
2051 }%
2052 {}%
2053 }%
2054 \ifnum\count@=0
2055 \GlossariesExtraWarningNoLine{Entry counting has been enabled
2056 \MessageBreak with \string\glenableentrycount\space but the
2057 \MessageBreak attribute 'entrycount' hasn't
2058 \MessageBreak been assigned to any of the defined
2059 \MessageBreak entries}%
2060 \fi
2061 }

```

trifcounttrigger

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

```

2062 \newcommand*\glxtrifcounttrigger[3]{%
2063 \glshasattribute{#1}{entrycount}%
2064 {%
2065 \ifnum\glentryprevcount{#1}>\glsetattribute{#1}{entrycount}\relax
2066 #3%
2067 \else
2068 #2%
2069 \fi
2070 }%
2071 {#3}%
2072 }

```

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

\@@cgl@

```

2073 \def\@@cgl@#1#2[#3]{%
2074 \glxtrifcounttrigger{#2}%
2075 {%
2076 \cglformat{#2}{#3}%
2077 \glunset{#2}%
2078 }%
2079 {%
2080 \@gl@{#1}{#2}[#3]%
2081 }%
2082 }%

```

\@@cglS@

```
2083 \def\@@cglSpl@#1#2[#3]{%
2084   \glSxtrifcounttrigger{#2}%
2085   {%
2086     \cglSplformat{#2}{#3}%
2087     \glSunset{#2}%
2088   }%
2089   {%
2090     \@glSpl@{#1}{#2}[#3]%
2091   }%
2092 }%
```

\@@cGls@

```
2093 \def\@@cGls@#1#2[#3]{%
2094   \glSxtrifcounttrigger{#2}%
2095   {%
2096     \cGlsformat{#2}{#3}%
2097     \glSunset{#2}%
2098   }%
2099   {%
2100     \@Gls@{#1}{#2}[#3]%
2101   }%
2102 }%
```

\@@cGlSpl@

```
2103 \def\@@cGlSpl@#1#2[#3]{%
2104   \glSxtrifcounttrigger{#2}%
2105   {%
2106     \cGlSplformat{#2}{#3}%
2107     \glSunset{#2}%
2108   }%
2109   {%
2110     \@GlSpl@{#1}{#2}[#3]%
2111   }%
2112 }%
```

\@@cGLS@

```
2113 \def\@@cGLS@#1#2[#3]{%
2114   \glSxtrifcounttrigger{#2}%
2115   {%
2116     \cGLSformat{#2}{#3}%
2117     \glSunset{#2}%
2118   }%
2119   {%
2120     \@GLS@{#1}{#2}[#3]%
2121   }%
2122 }%
```

\@@cGLSpl@

```

2123 \def\@cGLSpl@#1#2[#3]{%
2124   \glstrifcounttrigger{#2}%
2125   {%
2126     \cGLSplformat{#2}{#3}%
2127     \glset{#2}%
2128   }%
2129   {%
2130     \@GLSpl@{#1}{#2}[#3]%
2131   }%
2132 }%

```

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

```

\@cgl's@
2133 \def\@cgl's@#1#2[#3]{\@gl's@{#1}{#2}[#3]}

```

```

\@cGL's@
2134 \def\@cGL's@#1#2[#3]{\@GL's@{#1}{#2}[#3]}

```

```

\@cgl'spl@
2135 \def\@cgl'spl@#1#2[#3]{\@gl'spl@{#1}{#2}[#3]}

```

```

\@cGL'spl@
2136 \def\@cGL'spl@#1#2[#3]{\@GL'spl@{#1}{#2}[#3]}

```

Add all upper case versions not provided by glossaries.

```

\cGLS
2137 \newrobustcmd*{\cGLS}{\@gl's@hyp@opt\@cGLS}

```

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

```

2138 \newcommand*{\@cGLS}[2][{}]{%
2139   \new@ifnextchar[{\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2}[]}%
2140 }

```

```

\@cGLS@
2141 \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.

```

2142 \newcommand*{\cGLSformat}[2]{%
2143   \expandafter\mfirstucMakeUppercase\expandafter{\cgl'sformat{#1}{#2}}%
2144 }

```

```

\cGLSpl
2145 \newrobustcmd*{\cGLSpl}{\@gl's@hyp@opt\@cGLSpl}

```

\@cGLSp1 Defined the un-starred form. Need to determine if there is a final optional argument

```
2146 \newcommand*{\@cGLSp1}[2] [] {%
2147   \new@ifnextchar[{\@cGLSp1@{#1}{#2}}{\@cGLSp1@{#1}{#2} []}%
2148 }
```

\@cGLSp1@

```
2149 \def\@cGLSp1@#1#2[#3]{\@GLSp1@{#1}{#2}[#3]}
```

\cGLSplformat Format used by \@cGLSp1 if entry only used once on previous run. The first argument is the label, the second argument is the insert text.

```
2150 \newcommand*{\cGLSplformat}[2] {%
2151   \expandafter\mfirstuc\MakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
2152 }
```

Modify the trigger formats to check for the regular attribute.

\cglformat

```
2153 \renewcommand*{\cglformat}[2] {%
2154   \glsifregular{#1}
2155   {\glsentryfirst{#1}}%
2156   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
2157 }
```

\cGlsformat

```
2158 \renewcommand*{\cGlsformat}[2] {%
2159   \glsifregular{#1}
2160   {\Glsentryfirst{#1}}%
2161   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
2162 }
```

\cglsplformat

```
2163 \renewcommand*{\cglsplformat}[2] {%
2164   \glsifregular{#1}
2165   {\glsentryfirstplural{#1}}%
2166   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
2167 }
```

\cGlsplformat

```
2168 \renewcommand*{\cGlsplformat}[2] {%
2169   \glsifregular{#1}
2170   {\Glsentryfirstplural{#1}}%
2171   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}}%#2%
2172 }
```

New code similar to above for unit counting.

defunitcounters

```
2173 \newcommand*{\@@newglossaryentry@defunitcounters}{%
```

```

2174 \edef\@glo@countunit{\csuse{@glxstr@categoryattr@@\@glo@category @unitcount}}%
2175 \ifdefvoid\@glo@countunit
2176 {}%
2177 {%
2178   \@glxstr@ifunitcounter{\@glo@countunit}%
2179   }%
2180   {\expandafter\@glxstr@addunitcounter\expandafter{\@glo@countunit}}%
2181   }%
2182 }

```

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

```

2183 \newcommand*{\@glxstr@unitcountlist}{}

```

@addunitcounter

```

2184 \newcommand*{\@glxstr@addunitcounter}[1]{%
2185   \listadd{\@glxstr@unitcountlist}{#1}%
2186   \ifcsundef{glxstr@theunit@#1}
2187   {%
2188     \ifcsdef{theH#1}%
2189     {\csdef{glxstr@theunit@#1}{\csuse{theH#1}}}%
2190     {\csdef{glxstr@theunit@#1}{\csuse{the#1}}}%
2191   }%
2192   }%
2193 }

```

r@ifunitcounter

```

2194 \newcommand*{\@glxstr@ifunitcounter}[3]{%
2195   \xifinlist{#1}{\@glxstr@unitcountlist}{#2}{#3}%
2196 }

```

urrentunitcount

```

2197 \newcommand*\@glxstr@currentunitcount[1]{%
2198   glo@\glsetoklabel{#1}@currunit@\glgetattribute{#1}{unitcount}.%
2199   \csuse{glxstr@theunit@\glgetattribute{#1}{unitcount}}%
2200 }

```

eviousunitcount

```

2201 \newcommand*\@glxstr@previousunitcount[1]{%
2202   glo@\glsetoklabel{#1}@prevunit@\glgetattribute{#1}{unitcount}.%
2203   \csuse{glxstr@theunit@\glgetattribute{#1}{unitcount}}%
2204 }

```

t@currunitcount

```

2205 \newcommand*{\@gl@increment@currunitcount}[1]{%
2206   \glshasattribute{#1}{unitcount}%
2207   {%
2208     \edef\@glxstr@csname{\@glxstr@currentunitcount{#1}}%
2209     \ifcsundef{\@glxstr@csname}%

```

```

2210   {%
2211     \csgdef{\@glsxtr@csname}{1}%
2212     \listcsxadd
2213       {glo@\glsdetoklabel{#1}@unitlist}%
2214       {\glsgetattribute{#1}{unitcount}.%
2215        \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
2216     }%
2217   }%
2218   {%
2219     \csxdef{\@glsxtr@csname}%
2220     {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
2221   }%
2222 }%
2223 {}%
2224 }

```

t@currunitcount

```

2225 \newcommand*{\@gls@local@increment@currunitcount}[1]{%
2226   \gls@hasattribute{#1}{unitcount}%
2227   {%
2228     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
2229     \ifcsundef{\@glsxtr@csname}%
2230     {%
2231       \csdef{\@glsxtr@csname}{1}%
2232       \listcseadd
2233         {glo@\glsdetoklabel{#1}@unitlist}%
2234         {\glsgetattribute{#1}{unitcount}.%
2235          \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
2236       }%
2237     }%
2238     {%
2239       \csedef{\@glsxtr@csname}%
2240       {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
2241     }%
2242   }%
2243 {}%
2244 }

```

r@currunitcount

```

2245 \newcommand*{\@glsxtr@currunitcount}[2]{%
2246   \ifcsundef
2247     {glo@\glsdetoklabel{#1}@currunit@#2}%
2248     {0}%
2249     {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}%
2250 }%

```

r@prevunitcount

```

2251 \newcommand*{\@glsxtr@prevunitcount}[2]{%
2252   \ifcsundef

```

```

2253 {glo@\glsdetoklabel{#1}@prevunit@#2}%
2254 {0}%
2255 {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}%
2256 }%

```

entryunitcount

```

2257 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
2258   \appto@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
2259   \renewcommand*{\gls@defdocnewglossaryentry}{%
2260     \renewcommand*{\newglossaryentry[2]{%
2261       \PackageError{glossaries}{\string\newglossaryentry\space
2262       may only be used in the preamble when entry counting has
2263       been activated}{If you use \string\glsenableentryunitcount\space
2264       you must place all entry definitions in the preamble not in
2265       the document environment}%
2266     }%
2267   }%

```

New commands to access new fields:

```

2268   \newcommand*{\glsentrycurrcount}[1]{%
2269     \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2270     \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2271   }%
2272   \newcommand*{\glsentryprevcount}[1]{%
2273     \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2274     \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2275   }%

```

Access total count:

```

2276   \newcommand*{\glsentryprevtotalcount}[1]{%
2277     \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
2278     {0}%
2279     {%
2280       \number\csuse{glo@\glsdetoklabel{##1}@prevunittotal}
2281     }%
2282   }%

```

Access max value:

```

2283   \newcommand*{\glsentryprevmaxcount}[1]{%
2284     \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
2285     {0}%
2286     {%
2287       \number\csuse{glo@\glsdetoklabel{##1}@prevunitmax}
2288     }%
2289   }%

```

Adjust post unset and reset:

```

2290   \let@glsxtr@entryunitcount@org@unset\glsxtrpostunset

```

```

2291 \renewcommand*{\glxtrpostunset}[1]{%
2292   \@glxtr@entryunitcount@org@unset{##1}%
2293   \@glx@increment@currunitcount{##1}%
2294 }%
2295 \let\@glxtr@entryunitcount@org@localunset\glxtrpostlocalunset
2296 \renewcommand*{\glxtrpostlocalunset}[1]{%
2297   \@glxtr@entryunitcount@org@localunset{##1}%
2298   \@glx@local@increment@currunitcount{##1}%
2299 }%
2300 \let\@glxtr@entryunitcount@org@reset\glxtrpostreset
2301 \renewcommand*{\glxtrpostreset}[1]{%
2302   \glshasattribute{##1}{unitcount}%
2303   {%
2304     \edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
2305     \ifcsundef{\@glxtr@csname}%
2306     {%
2307       {\csgdef{\@glxtr@csname}{0}}%
2308     }%
2309   }%
2310 }%
2311 \let\@glxtr@entryunitcount@org@localreset\glxtrpostlocalreset
2312 \renewcommand*{\glxtrpostlocalreset}[1]{%
2313   \@glxtr@entryunitcount@org@localreset{##1}%
2314   \glshasattribute{##1}{unitcount}%
2315   {%
2316     \edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
2317     \ifcsundef{\@glxtr@csname}%
2318     {%
2319       {\csdef{\@glxtr@csname}{0}}%
2320     }%
2321   }%
2322 }%

```

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

```

2323 \let\@cgl@@@cgl@
2324 \let\@cgl@pl@@@cgl@pl@
2325 \let\@cGL@@@cGL@
2326 \let\@cGL@pl@@@cGL@pl@
2327 \let\@cGL@@@@cGL@
2328 \let\@cGL@pl@@@cGL@pl@

```

Write information to the aux file.

```

2329 \AtEndDocument{\@glx@write@entryunitcounts}%
2330 \renewcommand*{\@glx@entry@unitcount}[3]{%
2331   \csgdef{glo@\glx@detoklabel{##1}@prevunit@##3}{##2}%
2332   \ifcsundef{glo@\glx@detoklabel{##1}@prevunittotal}%
2333   {\csgdef{glo@\glx@detoklabel{##1}@prevunittotal}{##2}}%
2334   {%
2335     \csxdef{glo@\glx@detoklabel{##1}@prevunittotal}{

```

```

2336      \number\numexpr\csuse{glo@\glsdetoklabel{##1}@prevunittotal}+##2}%
2337  }%
2338  \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
2339  {\csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}}%
2340  {%
2341    \ifnum\csuse{glo@\glsdetoklabel{##1}@prevunitmax}<##2
2342    \csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}%
2343    \fi
2344  }%
2345  }%
2346  \let\glsenableentryunitcount\relax
2347  \renewcommand*{\glsenableentrycount}{%
2348    \PackageError{glossaries-extra}{\string\glsenableentrycount\space
2349      can't be used with \string\glsenableentryunitcount}%
2350    {Use one or other but not both commands}}%
2351  }%
2352 }
2353 \onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

2354 \newcommand*{\@gls@entry@unitcount}[3]{%

```

ryunitcounts@do

```

2355 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
2356   \immediate\write\@auxout
2357   {\string\@gls@entry@unitcount
2358     {\@glsentry}%
2359     {\@glsxtr@currunitcount{\@glsentry}{#1}%
2360     }%
2361     {#1}}%
2362 }

```

entryunitcounts

```

2363 \newcommand*{\@gls@write@entryunitcounts}{%
2364   \immediate\write\@auxout
2365   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
2366   \count@=0\relax
2367   \forallglsentries{\@glsentry}{%
2368     \gls@hasattribute{\@glsentry}{unitcount}%
2369     {%
2370       \ifglsused{\@glsentry}%
2371       {%
2372         \forlistcsloop
2373         {\@gls@write@entryunitcounts@do}%
2374         {glo@\glsdetoklabel{\@glsentry}@unitlist}%
2375       }%
2376       {}%
2377       \advance\count@ by \@ne
2378     }%

```

```

2379     {}%
2380 }%
2381 \ifnum\count@=0
2382   \GlossariesExtraWarningNoLine{Entry counting has been enabled
2383     \MessageBreak with \string\glsenableentryunitcount\space but the
2384     \MessageBreak attribute ‘unitcount’ hasn’t
2385     \MessageBreak been assigned to any of the defined
2386     \MessageBreak entries}%
2387 \fi
2388 }

```

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.

```

2389 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

2390 \glsenableentryunitcount

```

Redefine \gls etc:

```

2391 \renewcommand*{\gls}{\cglsl}%
2392 \renewcommand*{\Gls}{\cGls}%
2393 \renewcommand*{\glspl}{\cglspl}%
2394 \renewcommand*{\Glspl}{\cGlspl}%
2395 \renewcommand*{\GLS}{\cGLS}%
2396 \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

2397 \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

2398 \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
2399 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
2400   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
2401     can’t be used with \string\GlsXtrEnableEntryUnitCounting}%
2402   {Use one or other but not both commands}}%
2403 }

```

countunsetattr

```

2404 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
2405   \@for\@glsxtr@cat:=#1\do
2406   {%
2407     \ifdefempty{\@glsxtr@cat}{}%
2408     {%
2409       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
2410       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
2411     }%
2412   }%
2413 }

```

1.3.6 Acronym Modifications

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

`\SetGenericNewAcronym`

```

2414 \renewcommand*{\SetGenericNewAcronym}{%
2415   \let\@Gls@entryname\@Gls@acrentryname
2416   \renewcommand{\newacronym}[4][]{%
2417     \ifdefempty{\@glsacronymlists}%
2418     {%
2419       \def\@glo@type{\acronymtype}%
2420       \setkeys{glossentry}{##1}%
2421       \DeclareAcronymList{\@glo@type}%
2422     }%
2423   }%
2424   \glskeylisttok{##1}%
2425   \glslabeltok{##2}%
2426   \glsshorttok{##3}%
2427   \glslongtok{##4}%
2428   \newacronymhook
2429   \protected@edef\@do@newglossaryentry{%
2430     \noexpand\newglossaryentry{\the\glslabeltok}%
2431     {%
2432       type=\acronymtype,%
2433       name={\expandonce{\acronymentry{##2}}},%
2434       sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
2435       text={\the\glsshorttok},%
2436       short={\the\glsshorttok},%
2437       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
2438       long={\the\glslongtok},%
2439       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
2440       category=acronym,%
2441       \GenericAcronymFields,%
2442       \the\glskeylisttok
2443     }%
2444   }%
2445   \@do@newglossaryentry
2446 }%
2447 \renewcommand*{\acrfullfmt}[3]{%
2448   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}%
2449 \renewcommand*{\Acrfullfmt}[3]{%
2450   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}%
2451 \renewcommand*{\ACRfullfmt}[3]{%
2452   \glslink[##1]{##2}{%

```

```

2453     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
2454 \renewcommand*{\acrfullplfmt}[3]{%
2455     \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
2456 \renewcommand*{\Acrfullplfmt}[3]{%
2457     \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
2458 \renewcommand*{\ACRfullplfmt}[3]{%
2459     \glslink[##1]{##2}{%
2460         \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
2461 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
2462 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
2463 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
2464 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
2465 }

```

This will cause a problem for glossaries that contain a mixture of acronyms and abbreviations, so redefine `\newacronym` to use the new abbreviation interface.

First save the original definitions:

```

2466 \let\@glsxtr@org@setacronymstyle\setacronymstyle
2467 \let\@glsxtr@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`.

```

2468 \newcommand*{\MakeAcronymsAbbreviations}{%
2469     \renewcommand*{\newacronym}[4][]{%
2470         \newabbreviation[type=\acronymtype,category=acronym,##1]{##2}{##3}{##4}%
2471     }%
2472     \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
2473     \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%
2474     \renewcommand*{\setacronymstyle}[1]{%
2475         \PackageError{glossaries-extra}{\string\setacronymstyle{##1}
2476             unavailable.
2477             Use \string\setabbreviationstyle\space instead.
2478             The original acronym interface can be restored with
2479             \string\RestoreAcronyms}{}%
2480     }%
2481     \renewcommand*{\newacronymstyle}[1]{%
2482         \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
2483             available unless you restore the original acronym interface with
2484             \string\RestoreAcronyms}%
2485         \@glsxtr@org@newacronymstyle{##1}%
2486     }%
2487 }

```

Switch acronyms to abbreviations:

```

2488 \MakeAcronymsAbbreviations

```

RestoreAcronyms Restore acronyms to glossaries interface.

```

2489 \newcommand*\RestoreAcronyms}{%
2490   \SetGenericNewAcronym
2491   \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
2492   \renewcommand{\acronymfont}[1]{##1}%
2493   \let\setacronymstyle\@glxtr@org@setacronymstyle
2494   \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.

```

2495   \renewcommand*\@gls@link@checkfirsthyper{%
2496     \ifglused{\glslabel}%
2497     {\let\glxtrifwasfirstuse\@secondoftwo}
2498     {\let\glxtrifwasfirstuse\@firstoftwo}%
2499     \@glxtr@org@checkfirsthyper
2500   }
2501   \glssetcategoryattribute{acronym}{regular}{false}%
2502   \setacronymstyle{long-short}%
2503 }

```

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

```

2504 \renewcommand*\glsacspace}[1]{%
2505   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
2506   \ifdim\dimen@<\glsacspacemax~\else\space\fi
2507 }

```

`\glsacspacemax` Value used in the above.

```

2508 \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`.

`\reg@glosslist`

```

2509 \newcommand*\@glxtr@reg@glosslist{}

```

Save the original definition of `\makeglossaries`:

```

2510 \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`

```

2511 \renewcommand*\makeglossaries}[1][]{%
2512   \ifblank{#1}%

```

```

2513 {\@glxtr@org@makeglossaries}%
2514 {%
2515   \edef\@glxtr@reg@glosslist{#1}%
2516   \ifundef{\glswrite}{\newwrite\glswrite}{}%
2517   \protected@write\@auxout{}\string\providecommand
2518     \string\@glsorder[1]{}}
2519   \protected@write\@auxout{}\string\providecommand
2520     \string\@istfilename[1]{}}
2521   \protected@write\@auxout{}\string\@istfilename{\istfilename}}%
2522   \protected@write\@auxout{}\string\@glsorder{\@glsorder}}
2523   \protected@write\@auxout{}\string\glxtr@makeglossaries{#1}}
2524   \write\@auxout{\string\providecommand\string\@gls@reference[3]{}}%

```

Iterate through each supplied glossary type and activate it.

```

2525   \@for\@glo@type:=#1\do{%
2526     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
2527   }%

```

New glossaries must be created before \makeglossaries:

```

2528   \renewcommand*\newglossary[4][]{%
2529     \PackageError{glossaries}{New glossaries
2530       must be created before \string\makeglossaries}{You need
2531       to move \string\makeglossaries\space after all your
2532       \string\newglossary\space commands}}%

```

Any subsequent instances of this command should have no effect

```

2533   \let\@makeglossary\relax
2534   \let\makeglossary\relax
2535   \renewcommand\makeglossaries[1][]{}%

```

Disable all commands that have no effect after \makeglossaries

```

2536   \@disable@onlypremakeg

```

Allow see key:

```

2537   \let\gls@checkseeallowed\relax

```

Adjust \@do@seeglossary

```

2538   \let\@glxtr@org@doseeglossary\@do@seeglossary
2539   \renewcommand*\@do@seeglossary[2]{%
2540     \edef\@gls@label{\glsdetoklabel{##1}}%
2541     \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
2542     \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glxtr@reg@glosslist}%
2543     {\@glxtr@org@doseeglossary{##1}{##2}}%
2544     {%
2545       \protected@write\@auxout{}\string\@gls@reference
2546         {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
2547     }%
2548   }%
2549 }%
2550 }%

```

Adjust \@do@wrglossary

```

2551 \let\@glsxtr@do@@wrglossary\@do@@wrglossary
2552 \def\@do@@wrglossary{%
2553   \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
2554   \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
2555   {\@glsxtr@do@@wrglossary}%
2556   {\@gls@noidxglossary}%
2557 }%

```

Suppress warning about no \makeglossaries

```

2558 \let\warn@nomakeglossaries\relax
2559 \def\warn@noprintglossary{%
2560   \GlossariesWarningNoLine{No \string\printglossary\space
2561     or \string\printglossaries\space
2562     found.^^J(Remove \string\makeglossaries\space if you don't want
2563     any glossaries.)^^JThis document will not have a glossary}%
2564 }%

```

Only warn for glossaries not listed.

```

2565 \renewcommand{\@gls@noref@warn}[1]{%
2566   \edef\@gls@type{##1}%
2567   \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
2568   {%
2569     \GlossariesExtraWarning{Can't use
2570       \string\printnoidxglossary[type={\@gls@type}]
2571       when '\@gls@type' is listed in the optional argument of
2572       \string\makeglossaries}%
2573   }%
2574   {%
2575     \GlossariesWarning{Empty glossary for
2576       \string\printnoidxglossary[type={##1}].
2577       Rerun may be required (or you may have forgotten to use
2578       commands like \string\gls)}%
2579   }%
2580 }%

```

Adjust display number list to check for type:

```

2581 \renewcommand*{\glsdisplaynumberlist}[1]{%
2582   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2583   {\@glsxtr@idx@displaynumberlist{##1}}%
2584   {\@glsxtr@noidx@displaynumberlist{##1}}%
2585 }%

```

Adjust entry list:

```

2586 \renewcommand*{\glsentrynumberlist}[1]{%
2587   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2588   {\@glsxtr@idx@entrynumberlist{##1}}%
2589   {\@glsxtr@noidx@entrynumberlist{##1}}%
2590 }%

```

Adjust number list loop

```

2591 \renewcommand*{\glsnumberlistloop}[2]{%

```

```

2592 \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
2593 {%
2594 \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
2595 not available for glossary ‘##1’}{}%
2596 }%
2597 {\@glxtr@noidx@numberlistloop{##1}{##2}}%
2598 }%

```

Only sanitize sort for normal indexing glossaries.

```

2599 \renewcommand*\glsprestandardsort}[3]{%
2600 \expandafter\DTLifinlist\expandafter{##2}{\@glxtr@reg@glosslist}%
2601 {%
2602 \glsdosanitizesort
2603 }%
2604 {%
2605 \ifglssanitizesort
2606 \@gls@noidx@sanitizesort
2607 \else
2608 \@gls@noidx@nosanitizesort
2609 \fi
2610 }%
2611 }%

```

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

```

2612 \renewcommand*\new@glossaryentry[2]{%
2613 \PackageError{glossaries-extra}{Glossary entries must be defined
2614 in the preamble\MessageBreak when you use the optional argument
2615 of \string\makeglossaries}{Either move your definitions to the
2616 preamble or don't use the optional argument of
2617 \string\makeglossaries}%
2618 }%

```

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).

```

2619 \let\@glo@assign@sortkey\@glxtr@mixed@assign@sortkey
2620 \renewcommand*\@printgloss@setsort{%

```

Need to extract just the type value.

```

2621 \expandafter\@glxtr@gettype\expandafter,\@glxtr@printglossopts,%
2622 type=\glsdefaulttype,\@end@glxtr@gettype
2623 \def\@glo@sorttype{\@glo@default@sorttype}%
2624 }%

```

Check automake setting:

```

2625 \ifglsautomake
2626 \renewcommand*\@gls@doautomake{%
2627 \@for\@gls@type:=\@glxtr@reg@glosslist\do{%
2628 \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
2629 }%
2630 }%

```

```

2631     \fi
2632 }%
2633 }

```

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

`rgprintglossary` Save original definition (also needed for the on-the-fly macro).

```
2634 \let\@glxtr@orgprintglossary\@printglossary
```

`\@printglossary` Redefine.

```

2635 \renewcommand{\@printglossary}[2]{%
2636   \def\@glxtr@printglossopts{#1}%
2637   \@glxtr@orgprintglossary{#1}{#2}%
2638 }

```

`@makeglossaries` For the benefit of `makeglossaries`

```
2639 \newcommand*{\@glxtr@makeglossaries}[1]{}
```

`@glxtr@gettype` Get just the type.

```

2640 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
2641   \def\@glo@type{#2}%
2642 }

```

`@assign@sortkey` Assign the sort key.

```

2643 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
2644   \edef\@glo@type{\@glo@type}%
2645   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
2646   {%
2647     \@glo@no@assign@sortkey{#1}%
2648   }%
2649   {%
2650     \@glo@assign@sortkey{#1}%
2651   }%
2652 }%

```

Display number list for the regular version:

`splaynumberlist`

```
2653 \let\@glxtr@idx@displaynumberlist\glsdisplaynumberlist
```

Display number list for the “noidx” version:

`splaynumberlist`

```

2654 \newcommand*{\@glxtr@noidx@displaynumberlist}[1]{%
2655   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
2656   \ifdef\@gls@loclist
2657     {%
2658       \def\@gls@noidxloclist@sep{%

```

```

2659     \def\@gls@noidxloclist@sep{%
2660         \def\@gls@noidxloclist@sep{%
2661             \glsnumlistsep
2662         }%
2663     \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
2664 }%
2665 }%
2666 \def\@gls@noidxloclist@finalsep{}%
2667 \def\@gls@noidxloclist@prev{}%
2668 \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
2669 \@gls@noidxloclist@finalsep
2670 \@gls@noidxloclist@prev
2671 }%
2672 {%
2673 ??\glsdoifexists{#1}%
2674 {%
2675     \GlossariesWarning{Missing location list for ‘#1’. Either
2676         a rerun is required or you haven’t referenced the entry.}%
2677 }%
2678 }%
2679 }%
2680

```

And for the number list loop:

@numberlistloop

```

2681 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
2682     \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
2683     \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
2684     \let\@gls@org@glsseeformat\glsseeformat
2685     \let\glsnoidxdisplayloc#2\relax
2686     \let\glsseeformat#3\relax
2687     \ifdef\@gls@loclist
2688     {%
2689         \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
2690     }%
2691     {%
2692         ??\glsdoifexists{#1}%
2693         {%
2694             \GlossariesWarning{Missing location list for ‘##1’. Either
2695                 a rerun is required or you haven’t referenced the entry.}%
2696         }%
2697     }%
2698     \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
2699     \let\glsseeformat\@gls@org@glsseeformat
2700 }%

```

Same for entry number list.

entrynumberlist

```

2701 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
2702   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
2703   \ifdef\@gls@loclist
2704   {%
2705     \glsnoidxloclist{\@gls@loclist}%
2706   }%
2707   {%
2708     ??\glsdoifexists{#1}%
2709     {%
2710       \GlossariesWarning{Missing location list for ‘#1’. Either
2711         a rerun is required or you haven’t referenced the entry.}%
2712     }%
2713   }%
2714 }%

```

entrynumberlist

```

2715 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

@noidx@glossary

```

2716 \renewcommand*{\@print@noidx@glossary}{%
2717   \ifcsdef{\glsref@\@glo@type}%
2718   {%
2719     \ifcsdef{\@glo@sortmacro@\@glo@sorttype}%
2720     {%
2721       \csuse{\@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
2722     }%
2723     {%
2724       \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
2725     }%
2726     \glossarysection[\glossarytoctitle]{\glossarytitle}%
2727     \glossarypreamble

```

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

```

2728   \def\@gls@currentlettergroup{}%
2729   \begin{theglossary}%
2730   \glossaryheader
2731   \glsresetentrylist
2732   \forlistcsloop{\@gls@noidx@do}{\glsref@\@glo@type}%
2733   \end{theglossary}%
2734   \glossarypostamble
2735 }%
2736 {%

```

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

```

2737   \glsxtrifemptyglossary{\@glo@type}%
2738   {}%
2739   {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
2740   \@gls@noref@warn{\@glo@type}%

```

```

2741 }%
2742 }

```

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

@print@glossary

```

2743 \renewcommand{\@print@glossary}{%
2744   \makeatletter
2745   \@input@{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
2746   \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
2747   {}%
2748   {\glstrNoGlossaryWarning{\@glo@type}}%
2749   \ifglxindy
2750     \ifcsundef{@xdy@\@glo@type @language}%
2751     {%
2752       \edef\@do@auxoutstuff{%
2753         \noexpand\AtEndDocument{%
2754           \noexpand\immediate\noexpand\write\@auxout{%
2755             \string\providecommand\string\@xdylanguage[2]{}}%
2756           \noexpand\immediate\noexpand\write\@auxout{%
2757             \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
2758         }%
2759       }%
2760     }%
2761     {%
2762       \edef\@do@auxoutstuff{%
2763         \noexpand\AtEndDocument{%
2764           \noexpand\immediate\noexpand\write\@auxout{%
2765             \string\providecommand\string\@xdylanguage[2]{}}%
2766           \noexpand\immediate\noexpand\write\@auxout{%
2767             \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
2768               @language\endcsname}}%
2769         }%
2770       }%
2771     }%
2772     \@do@auxoutstuff
2773     \edef\@do@auxoutstuff{%
2774       \noexpand\AtEndDocument{%
2775         \noexpand\immediate\noexpand\write\@auxout{%
2776           \string\providecommand\string\@gls@codepage[2]{}}%
2777         \noexpand\immediate\noexpand\write\@auxout{%
2778           \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
2779       }%
2780     }%
2781     \@do@auxoutstuff
2782   \fi
2783   \renewcommand*{\@warn@nomakeglossaries}{%
2784     \GlossariesWarningNoLine{\string\makeglossaries\space
2785       hasn't been used,^^Jthe glossaries will not be updated}%

```

```

2786 }%
2787 }

```

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

`\GlsWarningHead` Header message.

```

2788 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
2789   This document is incomplete. The external file associated with
2790   the glossary ‘#1’ (which should be called \texttt{#2})
2791   hasn’t been created.%
2792 }

```

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

```

2793 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
2794   This has probably happened because there are no entries defined
2795   in this glossary.%
2796 }

```

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

```

2797 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
2798   If you don’t want this glossary,
2799   add \texttt{nomain} to your package option list when you load
2800   \texttt{glossaries-extra.sty}. For example:%
2801 }

```

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

```

2802 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
2803   Did you forget to use \texttt{type=#1} when you defined your
2804   entries? If you tried to load entries into this glossary with
2805   \texttt{\string\loadglsentries} did you remember to use
2806   \texttt{[#1]} as the optional argument? If you did, check that
2807   the definitions in the file you loaded all had the type set
2808   to \texttt{\string\glsdefaulttype}.%
2809 }

```

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

```

2810 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
2811   Check the contents of the file \texttt{#1}. If
2812   it’s empty, that means you haven’t indexed any of your entries in this
2813   glossary (using commands like \texttt{\string\gls} or
2814   \texttt{\string\glsadd}) so this list can’t be generated.
2815   If the file isn’t empty, the document build process hasn’t been
2816   completed.%
2817 }

```

`\GlsWarningAutoMake` Message when automake option has been used.

```

2818 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
2819   You may need to rerun \LaTeX. If you already have, it may be that
2820   \TeX’s shell escape doesn’t allow you to run

```

```

2821 \ifglxindy xindy\else makeindex\fi. Check the
2822 transcript file \texttt{\jobname.log}. If the shell escape is
2823 disabled, try one of the following:
2824
2825 \begin{itemize}
2826   \item Run the external (Lua) application:
2827
2828     \texttt{makeglossaries-lite.lua \string"\jobname\string"}
2829
2830   \item Run the external (Perl) application:
2831
2832     \texttt{makeglossaries \string"\jobname\string"}
2833 \end{itemize}
2834
2835 Then rerun \LaTeX\ on this document.
2836 \GlossariesExtraWarning{Rerun required to build the
2837 glossary ‘#1’ or check TeX’s shell escape allows
2838 you to run \ifglxindy xindy\else makeindex\fi}%
2839 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

2840 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
2841   You need to either replace \texttt{\string\makenoidxglossaries}
2842   with \texttt{\string\makeglossaries} or replace
2843   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
2844   \texttt{\string\printnoidxglossary}
2845   (or \texttt{\string\printnoidxglossaries}) and then rebuild
2846   this document.%
2847 }

```

WarningBuildInfo Build advice.

```

2848 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
2849   Try one of the following:
2850   \begin{itemize}
2851     \item Add \texttt{automake} to your package option list when you load
2852           \texttt{glossaries-extra.sty}. For example:
2853
2854           \texttt{\string\usepackage[automake]%
2855                 \glsopenbrace glossaries-extra\glsclosebrace}
2856
2857     \item Run the external (Lua) application:
2858
2859           \texttt{makeglossaries-lite.lua \string"\jobname\string"}
2860
2861     \item Run the external (Perl) application:
2862
2863           \texttt{makeglossaries \string"\jobname\string"}
2864   \end{itemize}

```

```

2865
2866 Then rerun \LaTeX\ on this document.%
2867 }

```

oGlsWarningTail Final paragraph.

```

2868 \newcommand{\GlsXtrNoGlsWarningTail}{%
2869 This message will be removed once the problem has been fixed.%
2870 }

```

GlsWarningNoOut No out file created. Build advice.

```

2871 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
2872 The file \texttt{#1} doesn't exist. This most likely means you haven't used
2873 \texttt{\string\makeglossaries} or you have used
2874 \texttt{\string\nofiles}. If this is just a draft version of the
2875 document, you can suppress this message using the
2876 \texttt{nomissingglstext} package option.%
2877 }

```

glossarywarning

```

2878 \newcommand*{@glsxtr@defaultnoglossarywarning}[1]{%
2879 \glossarysection[\glossarytoctitle]{\glossarytitle}
2880 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glo@type @in\endcsname}
2881 \par
2882 \glsxtrifemptyglossary{#1}%
2883 {%
2884 \GlsXtrNoGlsWarningEmptyStart\space
2885 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
2886 \medskip
2887 \noindent\texttt{\string\usepackage[nomain\ifglsacronym ,acronym\fi]%
2888 \glsopenbrace glossaries-extra\glsclosebrace}
2889 \medskip
2890 }%
2891 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
2892 }%
2893 {%
2894 \IfFileExists{\jobname.\csname @glo@type @out\endcsname}
2895 {%
2896 \GlsXtrNoGlsWarningCheckFile
2897 {\jobname.\csname @glo@type @out\endcsname}
2898
2899 \ifglsautomake
2900
2901 \GlsXtrNoGlsWarningAutoMake{#1}
2902
2903 \else
2904
2905 \ifthenelse{\equal{#1}{main}}%
2906 {%
2907 \GlsXtrNoGlsWarningEmptyMain\par

```

```

2908         \medskip
2909         \noindent\texttt{\string\usepackage[nomain]%
2910             \glsopenbrace glossaries-extra\glsclosebrace}
2911         \medskip
2912     }%
2913     {}%
2914
2915     \ifdefequal\makeglossaries\@no@makeglossaries
2916     {%
2917         \GlsXtrNoGlsWarningMisMatch
2918     }%
2919     {%
2920         \GlsXtrNoGlsWarningBuildInfo
2921     }%
2922 \fi
2923 }%
2924 {%
2925     \GlsXtrNoGlsWarningNoOut
2926     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
2927 }%
2928 }%
2929 \par
2930 \GlsXtrNoGlsWarningTail
2931 }

```

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

glsxtrresourcefile 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.

```

2932 \newcommand*{\glsxtrresourcefile}[2] [] {%
2933     \protected@write\@auxout{}\string\glsxtr@resource{#1}{#2}}%
2934     \glsxtr@writefields
2935     \let\@glsxtr@org@see@noindex\@gls@see@noindex
2936     \let\@gls@see@noindex\relax
2937     \InputIfFileExists{#2.glstex}{}%
2938     {%
2939         \GlossariesExtraWarning{No file '#2.glstex'}%
2940     }%
2941     \let\@gls@see@noindex\@glsxtr@org@see@noindex
2942 }
2943 \@onlypreamble\glsxtrresourcefile

```

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

```

2944 \newcommand*{\GlsXtrLoadResources}[1] [] {%
2945     \glsxtrresourcefile[#1]{\jobname}%
2946     \renewcommand*{\GlsXtrLoadResources}[1] [] {%
2947         \PackageError{glossaries-extra}%
2948         {Only 1 \string\GlsXtrLoadResources\space permitted per
2949         document. Use \string\glsxtrresourcefile\space for additional

```

```

2950     resources}%
2951     {}%
2952   }%
2953 }

glsxtr@resource
2954 \newcommand*{\glsxtr@resource}[2]{%

\glsxtr@fields
2955 \newcommand*{\glsxtr@fields}[1]{%

xtr@texencoding
2956 \newcommand*{\glsxtr@texencoding}[1]{%

tr@shortcutsval
2957 \newcommand*{\glsxtr@shortcutsval}[1]{%

sxtr@linkprefix
2958 \newcommand*{\glsxtr@linkprefix}[1]{%

xtr@writefields  This information only needs to be written once, so disable it after it's been used.
2959 \newcommand*{\glsxtr@writefields}{%
2960   \protected@write\@auxout{}\string\glsxtr@fields{\@gls@keymap}}%
2961   \ifdef\inputencodingname
2962   {%
2963     \protected@write\@auxout{}\string\glsxtr@texencoding{\inputencodingname}}%
2964   }%
2965   {%

    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.)

2966     \@ifpackageloaded{fontspec}%
2967     {\protected@write\@auxout{}\string\glsxtr@texencoding{utf8}}}%
2968     {}%
2969   }%
2970   \protected@write\@auxout{}\string\glsxtr@shortcutsval{\@glsxtr@shortcutsval}}%
2971   \protected@write\@auxout{}\string\glsxtr@linkprefix{\@glsxtr@linkprefix}}%
2972   \let\glsxtr@writefields\relax
2973 }

ntunsrtglossary  Similar to \printnoidxglossary but it displays all entries defined for the given glossary
                  without sorting.
2974 \newcommand*{\printunsrtglossary}[1][type=\glsdefaulttype]{%
2975   \@printglossary{#1}{\@print@unsrt@glossary}%
2976 }

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

```

```

2977 \newcommand*{\printunsrtglossaries}{%
2978   \foralllglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
2979 }

```

@unsrt@glossary

```

2980 \newcommand*{\@print@unsrt@glossary}{%
2981   \glossarysection[\glossarytoctitle]{\glossarytitle}%
2982   \glossarypreamble
      check for empty list
2983   \glxstrifemptyglossary{\@glo@type}%
2984   {%
2985     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
2986   }%
2987   {%
2988     \key@ifundefined{glossentry}{group}%
2989     {\let\@gls@getgrouptitle\@glxtr@noidx@getgrouptitle}%
2990     {\let\@gls@getgrouptitle\@glxtr@unsrt@getgrouptitle}%
2991     \begin{theglossary}%
2992     \glossaryheader
2993     \glsresetentrylist
2994     \def\@gls@currentlettergroup{}%
2995     \expandafter\@for\expandafter\glscurrententrylabel\expandafter
2996       :\expandafter=\csname glo@list@\@glo@type\endcsname\do{%
2997       \ifdefempty{\glscurrententrylabel}
2998       }%
2999       {\@glxtr@noidx@do\glscurrententrylabel}%
3000     }%
3001     \end{theglossary}%
3002   }%
3003   \glossarypostamble
3004 }

```

t@getgrouptitle

```

3005 \newcommand*{\@glxtr@unsrt@getgrouptitle}[2]{%
3006   \def#2{#1}%
3007 }

```

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

```

3008 \newcommand{\@glxtr@noidx@do}[1]{%
3009   \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3010   \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@location}%
3011   \ifglshasparent{#1}%
3012   {%
3013     \gls@level=\csuse{glo@\glsdetoklabel{#1}@level}\relax
3014     \ifdefvoid{\@gls@location}%
3015     {%
3016       \ifdefvoid{\@gls@loclist}%
3017     }%

```

```

3018     \subglossentry{\gls@level}{#1}{}%
3019 }%
3020 {%
3021     \subglossentry{\gls@level}{#1}%
3022     {%
3023         \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
3024     }%
3025 }%
3026 }%
3027 {%
3028     \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
3029 }%
3030 }%
3031 {%
3032     \letcs{\@gls@sort}{glo@glstdetoklabel{#1}@sort}%
3033     \key@ifundefined{glossentry}{group}%
3034     {%
3035         \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
3036     }%
3037     {%
3038         \protected@xdef\@glo@thislettergrp{%
3039             \csname glo@glstdetoklabel{#1}@group\endcsname}%
3040         }%
3041         \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
3042         {}%
3043         {%
3044             \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
3045             \gls@groupheading{\@glo@thislettergrp}%
3046         }%
3047         \let\@gls@currentlettergroup\@glo@thislettergrp
3048         \ifdefvoid{\@gls@location}%
3049         {%
3050             \ifdefvoid{\@gls@loclist}
3051             {%
3052                 \glossentry{#1}{}%
3053             }%
3054             {%
3055                 \glossentry{#1}%
3056                 {%
3057                     \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
3058                 }%
3059             }%
3060         }%
3061         {%
3062             \glossentry{#1}%
3063             {%
3064                 \glossaryentrynumbers{\@gls@location}%
3065             }%
3066         }%

```

```

3067 }%
3068 }

```

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.

```

3069 \@ifpackageloaded{glossaries-accsupp}
3070 {

```

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

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

```

3071 \newcommand*{\glsaccessname}[1]{%
3072   \glsnameaccessdisplay
3073   {%
3074     \glstryname{#1}%
3075   }%
3076   {#1}%
3077 }

```

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

```

3078 \newcommand*{\Glsaccessname}[1]{%
3079   \glsnameaccessdisplay
3080   {%
3081     \Glsstryname{#1}%
3082   }%
3083   {#1}%
3084 }

```

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

```

3085 \newcommand*{\GLSaccessname}[1]{%
3086   \glsnameaccessdisplay
3087   {%
3088     \mfirstucMakeUppercase{\glstryname{#1}}%
3089   }%
3090   {#1}%
3091 }

```

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

```

3092 \newcommand*{\glsaccesstext}[1]{%
3093   \glstextaccessdisplay
3094   {%
3095     \glstrytext{#1}%
3096   }%

```

```

3097     {#1}%
3098 }

```

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

```

3099 \newcommand*{\Glsaccesstext}[1]{%
3100   \glstextaccessdisplay
3101   {%
3102     \Glsentrytext{#1}%
3103   }%
3104   {#1}%
3105 }

```

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

```

3106 \newcommand*{\GLSaccesstext}[1]{%
3107   \glstextaccessdisplay
3108   {%
3109     \mfirstucMakeUppercase{\Glsentrytext{#1}}%
3110   }%
3111   {#1}%
3112 }

```

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

```

3113 \newcommand*{\glsaccessplural}[1]{%
3114   \glspluralaccessdisplay
3115   {%
3116     \glsentryplural{#1}%
3117   }%
3118   {#1}%
3119 }

```

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

```

3120 \newcommand*{\Glsaccessplural}[1]{%
3121   \glspluralaccessdisplay
3122   {%
3123     \Glsentryplural{#1}%
3124   }%
3125   {#1}%
3126 }

```

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

```

3127 \newcommand*{\GLSAccessplural}[1]{%
3128   \glspluralaccessdisplay
3129   {%
3130     \mfirstucMakeUppercase{\Glsentryplural{#1}}%
3131   }%
3132   {#1}%
3133 }

```

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

```
3134 \newcommand*{\glsaccessfirst}[1]{%
3135   \glsfirstaccessdisplay
3136   {%
3137     \glentryfirst{#1}%
3138   }%
3139   {#1}%
3140 }
```

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

```
3141 \newcommand*{\Glsaccessfirst}[1]{%
3142   \glsfirstaccessdisplay
3143   {%
3144     \Glentryfirst{#1}%
3145   }%
3146   {#1}%
3147 }
```

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

```
3148 \newcommand*{\GLSaccessfirst}[1]{%
3149   \glsfirstaccessdisplay
3150   {%
3151     \mfirstucMakeUppercase{\glentryfirst{#1}}%
3152   }%
3153   {#1}%
3154 }
```

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

```
3155 \newcommand*{\glsfirstplural}[1]{%
3156   \glsfirstpluralaccessdisplay
3157   {%
3158     \glentryfirstplural{#1}%
3159   }%
3160   {#1}%
3161 }
```

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

```
3162 \newcommand*{\Glsfirstplural}[1]{%
3163   \glsfirstpluralaccessdisplay
3164   {%
3165     \Glentryfirstplural{#1}%
3166   }%
3167   {#1}%
3168 }
```

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

```

3169 \newcommand*{\GLSaccessfirstplural}[1]{%
3170   \glfirstpluralaccessdisplay
3171   {%
3172     \mfirstucMakeUppercase{\glentryfirstplural{#1}}%
3173   }%
3174   {#1}%
3175 }

```

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

```

3176 \newcommand*{\glsaccesssymbol}[1]{%
3177   \glssymbolaccessdisplay
3178   {%
3179     \glentrysymbol{#1}%
3180   }%
3181   {#1}%
3182 }

```

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

```

3183 \newcommand*{\Glsaccesssymbol}[1]{%
3184   \glssymbolaccessdisplay
3185   {%
3186     \Glentrysymbol{#1}%
3187   }%
3188   {#1}%
3189 }

```

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

```

3190 \newcommand*{\GLSaccesssymbol}[1]{%
3191   \glssymbolaccessdisplay
3192   {%
3193     \mfirstucMakeUppercase{\glentrysymbol{#1}}%
3194   }%
3195   {#1}%
3196 }

```

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

```

3197 \newcommand*{\glsaccesssymbolplural}[1]{%
3198   \glssymbolpluralaccessdisplay
3199   {%
3200     \glentrysymbolplural{#1}%
3201   }%
3202   {#1}%
3203 }

```

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

```

3204 \newcommand*{\Glsaccesssymbolplural}[1]{%

```

```

3205 \glssymbolpluralaccessdisplay
3206 {%
3207 \Glsentrysymbolplural{#1}%
3208 }%
3209 {#1}%
3210 }

```

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

```

3211 \newcommand*{\GLSaccesssymbolplural}[1]{%
3212 \glssymbolpluralaccessdisplay
3213 {%
3214 \mfirstucMakeUppercase{\Glsentrysymbolplural{#1}}%
3215 }%
3216 {#1}%
3217 }

```

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

```

3218 \newcommand*{\glsaccessdesc}[1]{%
3219 \glsdescriptionaccessdisplay
3220 {%
3221 \glsentrydesc{#1}%
3222 }%
3223 {#1}%
3224 }

```

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

```

3225 \newcommand*{\Glsaccessdesc}[1]{%
3226 \glsdescriptionaccessdisplay
3227 {%
3228 \Glsentrydesc{#1}%
3229 }%
3230 {#1}%
3231 }

```

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

```

3232 \newcommand*{\GLSaccessdesc}[1]{%
3233 \glsdescriptionaccessdisplay
3234 {%
3235 \mfirstucMakeUppercase{\glsentrydesc{#1}}%
3236 }%
3237 {#1}%
3238 }

```

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

```

3239 \newcommand*{\glsaccessdescplural}[1]{%
3240 \glsdescriptionpluralaccessdisplay
3241 {%

```

```

3242     \glentrydescplural{#1}%
3243   }%
3244   {#1}%
3245 }

```

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

```

3246 \newcommand*\Glsaccessdescplural[1]{%
3247   \glsdescriptionpluralaccessdisplay
3248   {%
3249     \glentrydescplural{#1}%
3250   }%
3251   {#1}%
3252 }

```

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

```

3253 \newcommand*\GLSaccessdescplural[1]{%
3254   \glsdescriptionpluralaccessdisplay
3255   {%
3256     \mfirstucMakeUppercase{\glentrydescplural{#1}}%
3257   }%
3258   {#1}%
3259 }

```

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

```

3260 \newcommand*\glsaccessshort[1]{%
3261   \glsshortaccessdisplay
3262   {%
3263     \glentryshort{#1}%
3264   }%
3265   {#1}%
3266 }

```

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

```

3267 \newcommand*\GLSaccessshort[1]{%
3268   \glsshortaccessdisplay
3269   {%
3270     \glentryshort{#1}%
3271   }%
3272   {#1}%
3273 }

```

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

```

3274 \newcommand*\GLSAccessshort[1]{%
3275   \glsshortaccessdisplay
3276   {%
3277     \mfirstucMakeUppercase{\glentryshort{#1}}%

```

```

3278     }%
3279     {#1}%
3280 }

```

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

```

3281 \newcommand*{\saccessshortpl}[1]{%
3282   \glshortpluralaccessdisplay
3283   {%
3284     \glentryshortpl{#1}%
3285   }%
3286   {#1}%
3287 }

```

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

```

3288 \newcommand*{\Glsaccessshortpl}[1]{%
3289   \glshortpluralaccessdisplay
3290   {%
3291     \Glentryshortpl{#1}%
3292   }%
3293   {#1}%
3294 }

```

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

```

3295 \newcommand*{\GLSaccessshortpl}[1]{%
3296   \glshortpluralaccessdisplay
3297   {%
3298     \mfirstucMakeUppercase{\glentryshortpl{#1}}%
3299   }%
3300   {#1}%
3301 }

```

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

```

3302 \newcommand*{\glsaccesslong}[1]{%
3303   \glslongaccessdisplay{\glentrylong{#1}}{#1}%
3304 }

```

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

```

3305
3306 \newcommand*{\Glsaccesslong}[1]{%
3307   \glslongaccessdisplay{\Glentrylong{#1}}{#1}%
3308 }

```

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

```

3309 \newcommand*{\GLSaccesslong}[1]{%
3310   \glslongaccessdisplay
3311   {%
3312     \mfirstucMakeUppercase{\glentrylong{#1}}%

```

```

3313     }%
3314     {#1}%
3315 }

```

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

```

3316 \newcommand*{\glsaccesslongpl}[1]{%
3317     \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
3318 }

```

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

```

3319
3320 \newcommand*{\Glsaccesslongpl}[1]{%
3321     \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
3322 }

```

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

```

3323 \newcommand*{\GLSaccesslongpl}[1]{%
3324     \glslongpluralaccessdisplay
3325     {%
3326         \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
3327     }%
3328     {#1}%
3329 }

```

End of if part

```

3330 }
3331 {

```

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

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

```

3332 \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.

```

3333 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

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

```

3334 \newcommand*{\GLSaccessname}[1]{%
3335     \protect\mfirstucMakeUppercase{\glsentryname{#1}}}

```

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

```

3336 \newcommand*{\glsaccesstext}[1]{\glsentrytext{#1}}

```

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

```

3337 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}

```

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

```

3338 \newcommand*{\GLSaccessstext}[1]{%
3339 \protect\mfirstucMakeUppercase{\glstentrytext{#1}}}
```

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

```

3340 \newcommand*{\glsaccessplural}[1]{\glstentryplural{#1}}
```

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

```

3341 \newcommand*{\Glsaccessplural}[1]{\Glstentryplural{#1}}
```

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

```

3342 \newcommand*{\GLSaccessplural}[1]{%
3343 \protect\mfirstucMakeUppercase{\glstentryplural{#1}}}
```

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

```

3344 \newcommand*{\glsaccessfirst}[1]{\glstentryfirst{#1}}
```

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

```

3345 \newcommand*{\Glsaccessfirst}[1]{\Glstentryfirst{#1}}
```

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

```

3346 \newcommand*{\GLSaccessfirst}[1]{%
3347 \protect\mfirstucMakeUppercase{\glstentryfirst{#1}}}
```

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

```

3348 \newcommand*{\glsaccessfirstplural}[1]{\glstentryfirstplural{#1}}
```

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

```

3349 \newcommand*{\Glsaccessfirstplural}[1]{\Glstentryfirstplural{#1}}
```

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

```

3350 \newcommand*{\GLSaccessfirstplural}[1]{%
3351 \protect\mfirstucMakeUppercase{\glstentryfirstplural{#1}}}
```

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

```

3352 \newcommand*{\glsaccesssymbol}[1]{\glstentrysymbol{#1}}
```

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

```

3353 \newcommand*{\Glsaccesssymbol}[1]{\Glstentrysymbol{#1}}
```

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

```

3354 \newcommand*{\GLSaccesssymbol}[1]{%
3355 \protect\mfirstucMakeUppercase{\glstentrysymbol{#1}}}
```

esssymbolplural Display the symbolplural value (no link and no check for existence).
3356 \newcommand*{\glsaccesssymbolplural}[1]{\glsentrysymbolplural{#1}}

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.
3357 \newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}

esssymbolplural Display the symbolplural value (no link and no check for existence). converted to upper case.
3358 \newcommand*{\GLSaccesssymbolplural}[1]{%
3359 \protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}

\glsaccessdesc Display the desc value (no link and no check for existence).
3360 \newcommand*{\glsaccessdesc}[1]{\glsentrydesc{#1}}

\Glsaccessdesc Display the desc value (no link and no check for existence) with the first letter converted to upper case.
3361 \newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}

\GLSaccessdesc Display the desc value (no link and no check for existence). converted to upper case.
3362 \newcommand*{\GLSaccessdesc}[1]{%
3363 \protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}

ccessdescplural Display the descplural value (no link and no check for existence).
3364 \newcommand*{\glsaccessdescplural}[1]{\glsentrydescplural{#1}}

ccessdescplural Display the descplural value (no link and no check for existence) with the first letter converted to upper case.
3365 \newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}

ccessdescplural Display the descplural value (no link and no check for existence). converted to upper case.
3366 \newcommand*{\GLSaccessdescplural}[1]{%
3367 \protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}

\glsaccessshort Display the short form (no link and no check for existence).
3368 \newcommand*{\glsaccessshort}[1]{\glsentryshort{#1}}

\Glsaccessshort Display the short form with first letter converted to uppercase (no link and no check for existence).
3369 \newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}

\GLSaccessshort Display the short value (no link and no check for existence). converted to upper case.
3370 \newcommand*{\GLSaccessshort}[1]{%
3371 \protect\mfirstucMakeUppercase{\glsentryshort{#1}}}

lsaccessshortpl Display the short plural form (no link and no check for existence).
3372 \newcommand*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}

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

```
3373 \newcommand*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

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

```
3374 \newcommand*{\GLSaccessshortpl}[1]{%
```

```
3375 \protect\mfirstucMakeUppercase{\glentryshortpl{#1}}}
```

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

```
3376 \newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}
```

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

```
3377 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

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

```
3378 \newcommand*{\GLSaccesslong}[1]{%
```

```
3379 \protect\mfirstucMakeUppercase{\glentrylong{#1}}}
```

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

```
3380 \newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}
```

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

```
3381 \newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

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

```
3382 \newcommand*{\GLSaccesslongpl}[1]{%
```

```
3383 \protect\mfirstucMakeUppercase{\glentrylongpl{#1}}}
```

End of else part

```
3384 }
```

1.5 Categories

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

```
3385 \glsaddstoragekey{category}{general}{\glscategory}
```

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

```
3386 \newcommand{\glsifcategory}[4]{%
```

```
3387 \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
```

```
3388 }
```

Categories can have attributes.

categoryattribute

`\glissetcategoryattribute{<category>}{<attribute-label>}{<value>}`

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

```
3389 \newcommand*{\glissetcategoryattribute}[3]{%
3390   \csdef{@glsxtr@categoryattr@@#1@#2}{#3}%
3391 }
```

categoryattribute

`\glsggetcategoryattribute{<category>}{<attribute-label>}`

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

```
3392 \newcommand*{\glsggetcategoryattribute}[2]{%
3393   \csuse{@glsxtr@categoryattr@@#1@#2}%
3394 }
```

categoryattribute

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

Tests if the category has the given attribute set.

```
3395 \newcommand*{\glshascategoryattribute}[4]{%
3396   \ifcsvoid{@glsxtr@categoryattr@@#1@#2}{#4}{#3}%
3397 }
```

\glissetattribute

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

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

```
3398 \newcommand*{\glissetattribute}[3]{%
3399   \glissetcategoryattribute{\glscategory{#1}}{#2}{#3}%
3400 }
```

\glsggetattribute

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

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

```
3401 \newcommand*{\glsggetattribute}[2]{%
3402   \glsggetcategoryattribute{\glscategory{#1}}{#2}%
3403 }
```

`\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.

```
3404 \newcommand*{\glshasattribute}[4]{%
3405   \ifglentryexists{#1}%
3406   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
3407   {#4}%
3408 }
```

`categoryattribute` `\glcifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

True if category has the attribute with the given value.

```
3409 \newcommand{\glcifcategoryattribute}[5]{%
3410   \ifcsundef{@glxtr@categoryattr@#1@#2}%
3411   {#5}%
3412   {\ifcsstring{@glxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
3413 }
```

`\glcifattribute` `\glcifattribute{<entry label>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

Short cut to determine if the given entry has a category with the given attribute set.

```
3414 \newcommand{\glcifattribute}[5]{%
3415   \ifglentryexists{#1}%
3416   {\glcifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
3417   {#5}%
3418 }
```

Set attributes for the default general category:

```
3419 \glsetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
3420 \glsetcategoryattribute{acronym}{regular}{true}
```

`regularcategory` Convenient shortcut to create add the regular attribute.

```
3421 \newcommand*{\glsetregularcategory}[1]{%
3422   \glsetcategoryattribute{#1}{regular}{true}%
3423 }
```

`\regularcategory` `\glsifregularcategory{<category>}{<true part>}{<>false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```
3424 \newcommand{\glsifregularcategory}[3]{%
3425   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
3426 }
```

`\notregularcategory` `\glsifnotregularcategory{<category>}{<true part>}{<>false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```
3427 \newcommand{\glsifnotregularcategory}[3]{%
3428   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
3429 }
```

`\glsifregular` `\glsifregular{<entry label>}{<true part>}{<>false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
3430 \newcommand{\glsifregular}[3]{%
3431   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
3432 }
```

`\glsifnotregular` `\glsifnotregular{<entry label>}{<true part>}{<>false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
3433 \newcommand{\glsifnotregular}[3]{%
3434   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
3435 }
```

`\foreachincategory` `\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.

```
3436 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
```

```

3437 \forallglossaries[#1]{#3}%
3438 {%
3439     \forallsentries[#3]{#4}%
3440     {%
3441         \glsifcategory{#4}{#2}{#5}{}%
3442     }%
3443 }%
3444 }

```

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.

```

3445 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
3446     \forallglossaries[#1]{#4}%
3447     {%
3448         \forallsentries[#4]{#5}%
3449         {%
3450             \glsifattribute{#5}{#2}{#3}{#6}{}%
3451         }%
3452     }%
3453 }

```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to index and add `\glstrpostdescription`.

```

3454 \ifdef\newterm
3455 {%

```

\newterm

```

3456     \renewcommand*{\newterm}[2][ ]{%
3457         \newglossaryentry{#2}%
3458         {type=index,category=index,name={#2},%
3459          description={\glstrpostdescription\nopostdesc},#1}%
3460     }

```

Indexed terms are regular by default.

```

3461     \glssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

3462     \newcommand*{\glstrpostdescindex}{}

3463 }
3464 {}

```

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.

```
3465 \ifdef\printsymbols
3466 {%
```

`glsxtrnewsymbol` Unlike `\newterm`, this has a separate argument for the label (since the symbol will likely contain commands).

```
3467 \newcommand*{\glsxtrnewsymbol}[3] [] {%
3468 \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
3469 }
```

Symbols are regular by default.

```
3470 \glsssetcategoryattribute{symbol}{regular}{true}
```

`rpostdescsymbol`

```
3471 \newcommand*{\glsxtrpostdescsymbol}{}
3472 }
3473 {}
```

Similar for the numbers option.

```
3474 \ifdef\printnumbers
3475 {%
```

`glsxtrnewnumber`

```
3476 \ifdef\printnumbers
3477 \newcommand*{\glsxtrnewnumber}[3] [] {%
3478 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
3479 }
```

Numbers are regular by default.

```
3480 \glsssetcategoryattribute{number}{regular}{true}
```

`rpostdescnumber`

```
3481 \newcommand*{\glsxtrpostdescnumber}{}
3482 }
3483 {}
```

`glsxtrsetcategory` Set the category for all listed labels. The first argument is the list of entry labels and the second argument is the category label.

```
3484 \newcommand*{\glsxtrsetcategory}[2] {%
3485 \@for\@glsxtr@label:=#1\do
3486 {%
3487 \glsfieldxddef{\@glsxtr@label}{category}{#2}%
3488 }%
3489 }
```

`\categoryforall` Set the category for all entries in the listed glossaries. The first argument is the list of glossary labels and the second argument is the category label.

```

3490 \newcommand*{\glxtrsetcategoryforall}[2]{%
3491   \forallglossaries[#1]{\@glxtr@type}{%
3492     \forallsentries[\@glxtr@type]{\@glxtr@label}%
3493     {%
3494       \glsexdef{\@glxtr@label}{category}{#2}%
3495     }%
3496   }%
3497 }

```

`\glxtrfieldtitlecase` `\glxtrfieldtitlecase{\langle label \rangle}{\langle field \rangle}`

Apply title casing to the contents of the given field.

```

3498 \newcommand*{\glxtrfieldtitlecase}[2]{%
3499   \expandafter\glxtrfieldtitlecasesecs\expandafter
3500   {\csname glo@\glxtrdetoklabel{#1}@#2\endcsname}%
3501 }

```

`\glxtrfieldtitlecasesecs` The command used by `\glxtrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```

3502 \newcommand*{\glxtrfieldtitlecasesecs}[1]{\xcapitalisewords{#1}}

```

Provide a convenient way to modify glossary styles without having to define a new style just to convert the first letter of fields to upper case.

`\glossentrydesc` If the `glossdesc` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

3503 \ifpackageloaded{glossaries-accsupp}
3504 {
3505   \renewcommand*{\glossentrydesc}[1]{%
3506     \glxdoifexistsorwarn{#1}%
3507     {%
3508       \glxsetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossdescfont` attribute to determine the font applied.

```

3509     \glshasattribute{#1}{glossdescfont}%
3510     {%
3511       \edef\@glxtr@attrval{\glxgetattribute{#1}{glossdescfont}}%
3512       \ifcsdef{\@glxtr@attrval}%
3513       {%
3514         \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
3515       }%
3516       {%
3517         \GlossariesExtraWarning{Unknown control sequence name
3518           '\@glxtr@attrval' supplied in glossdescfont attribute

```

```

3519         for entry '#1'. Ignoring}%
3520         \let\@glxtr@glossdescfont\@firstofone
3521     }%
3522 }%
3523 {\let\@glxtr@glossdescfont\@firstofone}%
3524 \glsifattribute{#1}{glossdesc}{firstuc}%
3525 {%
3526     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
3527 }%
3528 {%
3529     \glsifattribute{#1}{glossdesc}{title}%
3530 {%
3531         \@glxtr@do@titlecaps@warn
3532         \glsdescriptionaccessdisplay
3533         {%
3534             \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
3535         }%
3536         {#1}%
3537     }%
3538     {%
3539         \@glxtr@glossdescfont{\glsaccessdesc{#1}}%
3540     }%
3541 }%
3542 }%
3543 }
3544 }
3545 {
3546 \renewcommand*{\glossentrydesc}[1]{%
3547     \glsdoifexistsorwarn{#1}%
3548     {%
3549         \glssetabbrvfmt{\glscategory{#1}}%
3550         \glsattribute{#1}{glossdescfont}%
3551         {%
3552             \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
3553             \ifcsdef{\@glxtr@attrval}%
3554             {%
3555                 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
3556             }%
3557             {%
3558                 \GlossariesExtraWarning{Unknown control sequence name
3559                     '\@glxtr@attrval' supplied in glossdescfont attribute
3560                     for entry '#1'. Ignoring}%
3561                 \let\@glxtr@glossdescfont\@firstofone
3562             }%
3563         }%
3564         {\let\@glxtr@glossdescfont\@firstofone}%
3565         \glsifattribute{#1}{glossdesc}{firstuc}%
3566         {%
3567             \@glxtr@glossdescfont{\Glsentrydesc{#1}}%

```

```

3568 }%
3569 {%
3570   \glsifattribute{#1}{glossdesc}{title}%
3571   {%
3572     \@glsxtr@do@titlecaps@warn
3573     \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
3574   }%
3575   {%
3576     \@glsxtr@glossdescfont{\glsentrydesc{#1}}%
3577   }%
3578 }%
3579 }%
3580 }
3581 }

```

`\glossentryname` If the `glossname` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

3582 \ifpackageloaded{glossaries-accsupp}
3583 {
3584   \renewcommand*{\glossentryname}[1]{%
3585     \@glsdoifexistsorwarn{#1}%
3586     {%
3587       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```

3588   \glsifattribute{#1}{glossnamefont}%
3589   {%
3590     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
3591     \ifcsdef{\@glsxtr@attrval}%
3592     {%
3593       \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
3594     }%
3595     {%
3596       \GlossariesExtraWarning{Unknown control sequence name
3597         ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
3598         for entry ‘#1’. Reverting to default \string\glsnamefont}%
3599       \let\@glsxtr@glossnamefont\glsnamefont
3600     }%
3601   }%
3602   {\let\@glsxtr@glossnamefont\glsnamefont}%
3603   \glsifattribute{#1}{glossname}{firstuc}%
3604   {%
3605     \glsnameaccessdisplay
3606     {%
3607       \@glsxtr@glossnamefont{\Glsentryname{#1}}%
3608     }%
3609     {#1}%
3610   }%
3611   {%
3612     \glsifattribute{#1}{glossname}{title}%

```

```

3613      {%
3614      \@glstr@do@titlecaps@warn
3615      \glsnameaccessdisplay
3616      {%
3617      \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}}%
3618      }%
3619      {#1}%
3620      }%
3621      {%
3622      \glsifattribute{#1}{glossname}{uc}%
3623      {%
3624      \glsnameaccessdisplay
3625      {%

```

Hide the label from the upper-casing command.

```

3626      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
3627      \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
3628      }%
3629      {#1}%
3630      }%
3631      {%
3632      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
3633      \glsnameaccessdisplay
3634      {%
3635      \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
3636      }%
3637      {#1}%
3638      }%
3639      }%
3640      }%

```

Do post-name hook:

```

3641      \glstrpostnamehook{#1}%
3642      }%
3643  }
3644 }
3645 {
3646 \renewcommand*{\glossentryname}[1]{%
3647 \@glsdoifexistsorwarn{#1}%
3648 {%
3649 \glssetabbrvfmt{\glscategory{#1}}}%
3650 \glsattribute{#1}{glossnamefont}%
3651 {%
3652 \edef\@glstr@attrval{\glsattribute{#1}{glossnamefont}}%
3653 \ifcsdef{\@glstr@attrval}%
3654 {%
3655 \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
3656 }%
3657 {%
3658 \GlossariesExtraWarning{Unknown control sequence name

```

```

3659         ‘\@glxtr@attrval’ supplied in glossnamefont attribute
3660         for entry ‘#1’. Reverting to default \string\glsnamefont}%
3661         \let\@glxtr@glossnamefont\glsnamefont
3662     }%
3663 }%
3664 {\let\@glxtr@glossnamefont\glsnamefont}%
3665 \glsifattribute{#1}{glossname}{firstuc}%
3666 {%
3667     \@glxtr@glossnamefont{\Glsentryname{#1}}%
3668 }%
3669 {%
3670     \glsifattribute{#1}{glossname}{title}%
3671     {%
3672         \@glxtr@do@titlecaps@warn
3673         \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
3674     }%
3675     {%
3676         \glsifattribute{#1}{glossname}{uc}%
3677         {%

```

Hide the label from the upper-casing command.

```

3678         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
3679         \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
3680     }%
3681     {%

```

This little trick is used by glossaries to allow the user to redefine \glsnamefont to use \makefirstuc. Support it even though they can now use the firstuc attribute.

```

3682         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
3683         \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
3684     }%
3685 }%
3686 }%

```

Do post-name hook.

```

3687     \glxtrpostnamehook{#1}%
3688 }%
3689 }
3690 }

```

\Glossentryname Redefine to set the abbreviation format and accessibility support.

```

3691 \@ifpackageloaded{glossaries-accsupp}
3692 {
3693     \renewcommand*{\Glossentryname}[1]{%
3694         \@glstdoifexistsorwarn{#1}%
3695         {%
3696             \glsetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

3697         \glshasattribute{#1}{glossnamefont}%
3698         {%

```

```

3699     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
3700     \ifcsdef{\@glsxtr@attrval}%
3701     {%
3702         \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
3703     }%
3704     {%
3705         \GlossariesExtraWarning{Unknown control sequence name
3706             '\@glsxtr@attrval' supplied in glossnamefont attribute
3707             for entry '#1'. Reverting to default \string\glsnamefont}%
3708         \let\@glsxtr@glossnamefont\glsnamefont
3709     }%
3710 }%
3711 {\let\@glsxtr@glossnamefont\glsnamefont}%
3712 \glsnameaccessdisplay
3713 {%
3714     \@glsxtr@glossnamefont{\Glsentryname{#1}}%
3715 }%
3716 {#1}%

```

Do post-name hook:

```

3717     \glsxtrpostnamehook{#1}%
3718 }%
3719 }
3720 }
3721 {
3722     \renewcommand*{\Glossentryname}[1]{%
3723         \@glsdoifexistsorwarn{#1}%
3724         {%
3725             \glssetabbrvfmt{\glscategory{#1}}%
3726             \glschasattribute{#1}{glossnamefont}%
3727             {%
3728                 \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
3729                 \ifcsdef{\@glsxtr@attrval}%
3730                 {%
3731                     \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
3732                 }%
3733                 {%
3734                     \GlossariesExtraWarning{Unknown control sequence name
3735                         '\@glsxtr@attrval' supplied in glossnamefont attribute
3736                         for entry '#1'. Reverting to default \string\glsnamefont}%
3737                     \let\@glsxtr@glossnamefont\glsnamefont
3738                 }%
3739             }%
3740             {\let\@glsxtr@glossnamefont\glsnamefont}%
3741             \@glsxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

3742     \glsxtrpostnamehook{#1}%
3743 }%
3744 }

```

3745 }

Provide a convenient way to also index the entries using the standard `\index` mechanism. This may use different actual, encap and escape characters to those used for the glossaries.

`glstrpostnamehook` Hook to append stuff after the name is displayed in the glossary. The argument is the entry's label.

```
3746 \newcommand*{\glstrpostnamehook}[1]{%
3747   \def\@glstrnumberformat{\glstrnumberformat}%
3748   \glstrdoautoindexname{#1}{\indexname}%
```

Allow categories to hook in here.

```
3749   \csuse{\glstrpostname\glstrcategory{\glstrcurrententrylabel}}%
3750 }
```

`glstrformat@override` Determines if the format key should override the indexing attribute value.

```
3751 \newif\if@glstr@format@override
3752 \@glstr@format@overridefalse
```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`glstrFormatOverride`

```
3753 \@ifpackageloaded{hyperref}{
3754 {
```

If `hyperref`'s `hyperindex` option is on, then `hyperref` will automatically add `\hyperpage`, so don't add it.

```
3755   \ifHy@hyperindex
3756     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
3757       \@glstr@format@override true
3758       \appto\theindex{\let\glshypernumber\@firstofone}%
3759     }
3760   \else
3761     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
3762       \@glstr@format@override true
3763       \appto\theindex{\let\glshypernumber\hyperpage}%
3764     }
3765   \fi
3766 }
3767 {
3768   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
3769     \@glstr@format@override true
3770   }
3771 }
3772 \@onlypreamble\GlsXtrEnableIndexFormatOverride
```

`glstrdoautoindexname`

```
3773 \newcommand*{\glstrdoautoindexname}[2]{%
3774   \glshasattribute{#1}{#2}%
3775   {%
```

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.

```
3776 \glxtr@autoindex@setname{#1}%
```

If the attribute value is simply “true” don't add an encap, otherwise use the value as the encap.

```
3777 \protected@edef\glxtr@attrval{\glxtr@getattribute{#1}{#2}}%
3778 \ifglxtr@format@override
3779 \ifdefstring{\glxtr@numberformat}{\glxtr@numberformat}%
3780 {\let\glxtr@attrval\glxtr@numberformat}%
3781 \fi
3782 \ifdefstring{\glxtr@attrval}{true}%
3783 {}%
3784 {\eappto\glo@name{\glxtr@autoindex@encap\glxtr@attrval}}%
3785 \expandafter\index\expandafter{\glo@name}%
3786 }%
3787 {}%
3788 }
```

toindex@setname Assign \glo@name for use with indexname attribute.

```
3789 \newcommand*{\glxtr@autoindex@setname}[1]{%
3790 \def\glo@name{\string\glxtr@entryname{#1}}%
3791 \glxtr@entryfield{\glo@sort}{#1}{sort}%
3792 \glxtr@checkmkidxchars\glo@sort
3793 \glxtr@autoindex@doextra@esc\glo@sort
3794 \epreto\glo@name{\glo@sort\glxtr@autoindex@at}%
3795 }
```

dex@doextra@esc

```
3796 \newcommand*{\glxtr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
3797 \ifx\glxtr@autoindex@esc\glxtr@quotechar
3798 \else
3799 \def\glxtr@checkedmkidx{}%
3800 \edef\glxtr@checkspch{%
3801 \noexpand\glxtr@autoindex@escquote\expandonce{#1}%
3802 \noexpand\@empty\glxtr@autoindex@esc\noexpand\@nnil
3803 \glxtr@autoindex@esc\noexpand\@empty\noexpand\glxtr@endescspch}%
3804 \@glxtr@checkspch
3805 \let#1\glxtr@checkedmkidx\relax
3806 \fi
```

Escape actual character unless it has already been escaped.

```
3807 \ifx\glxtr@autoindex@at\glxtr@actualchar
3808 \else
3809 \def\glxtr@checkedmkidx{}%
3810 \edef\glxtr@checkspch{%
3811 \noexpand\glxtr@autoindex@escat\expandonce{#1}%
3812 \noexpand\@empty\glxtr@autoindex@at\noexpand\@nnil
3813 \glxtr@autoindex@at\noexpand\@empty\noexpand\glxtr@endescspch}%

```

```

3814 \@@glxtr@checkspch
3815 \let#1\@gls@checkedmkidx\relax
3816 \fi

```

Escape level character unless it has already been escaped.

```

3817 \ifx\@glxtr@autoindex@level\@gls@levelchar
3818 \else
3819 \def\@gls@checkedmkidx{}%
3820 \edef\@@glxtr@checkspch{%
3821 \noexpand\@glxtr@autoindex@esclevel\expandonce{#1}%
3822 \noexpand\@empty\@glxtr@autoindex@level\noexpand\@nnil
3823 \@glxtr@autoindex@level\noexpand\@empty\noexpand\@glxtr@endescspch}%
3824 \@@glxtr@checkspch
3825 \let#1\@gls@checkedmkidx\relax
3826 \fi

```

Escape encap character unless it has already been escaped.

```

3827 \ifx\@glxtr@autoindex@encap\@gls@encapchar
3828 \else
3829 \def\@gls@checkedmkidx{}%
3830 \edef\@@glxtr@checkspch{%
3831 \noexpand\@glxtr@autoindex@escencap\expandonce{#1}%
3832 \noexpand\@empty\@glxtr@autoindex@encap\noexpand\@nnil
3833 \@glxtr@autoindex@encap\noexpand\@empty\noexpand\@glxtr@endescspch}%
3834 \@@glxtr@checkspch
3835 \let#1\@gls@checkedmkidx\relax
3836 \fi
3837 }

```

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.

```

3838 \newcommand*{\@glxtr@autoindex@at}{}

```

trSetActualChar Set the actual character.

```

3839 \newcommand*{\GlsXtrSetActualChar}[1]{%
3840 \gdef\@glxtr@autoindex@at{#1}%
3841 \def\@glxtr@autoindex@escat##1#1##2#1##3\@glxtr@endescspch{%
3842 \@@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escat}{##1}{##2}{##3}%
3843 }%
3844 }
3845 \onlypreamble\GlsXtrSetActualChar
3846 \makeatother
3847 \GlsXtrSetActualChar{@}
3848 \makeatletter

```

autoindex@encap Encap character for use with \index.

```

3849 \newcommand*{\@glxtr@autoindex@encap}{}

```

XtrSetEncapChar Set the encap character.

```
3850 \newcommand*{\GlsXtrSetEncapChar}[1]{%
3851   \gdef\@glstr@autoindex@encap{#1}%
3852   \def\@glstr@autoindex@escencap##1##2##3\@glstr@endescspch{%
3853     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escencap}{##1}{##2}{##3}%
3854   }%
3855 }
3856 \GlsXtrSetEncapChar{||}
3857 \@onlypreamble\GlsXtrSetEncapChar
```

autoindex@level Level character for use with \index.

```
3858 \newcommand*{\@glstr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
3859 \newcommand*{\GlsXtrSetLevelChar}[1]{%
3860   \gdef\@glstr@autoindex@level{#1}%
3861   \def\@glstr@autoindex@esclevel##1##2##3\@glstr@endescspch{%
3862     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@esclevel}{##1}{##2}{##3}%
3863   }%
3864 }
3865 \GlsXtrSetLevelChar{!}
3866 \@onlypreamble\GlsXtrSetLevelChar
```

r@autoindex@esc Escape character for use with \index.

```
3867 \newcommand*{\@glstr@autoindex@esc}{"}
```

lsXtrSetEscChar Set the escape character.

```
3868 \newcommand*{\GlsXtrSetEscChar}[1]{%
3869   \gdef\@glstr@autoindex@esc{#1}%
3870   \def\@glstr@autoindex@escquote##1##2##3\@glstr@endescspch{%
3871     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escquote}{##1}{##2}{##3}%
3872   }%
3873 }
3874 \GlsXtrSetEscChar{"}
3875 \@onlypreamble\GlsXtrSetEscChar
```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```
3876 \ifdef\actualchar
3877   {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
3878 }
```

Quote character \quotechar:

```
3879 \ifdef\quotechar
3880   {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
3881 }
```

Level character \levelchar:

```
3882 \ifdef\levelchar
3883   {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
3884 }
```

Encap character \encapchar:

```
3885 \ifdef\encapchar
3886 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
3887 {}
```

leto@endescspch

```
3888 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

toindex@esc@spch

```
\@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}
```

```
3889 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
3890   \@glstmpb=\expandafter{\@glsccheckedmkidx}%
3891   \toks@={#3}%
3892   \ifx\@nnil#3\relax
3893     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
3894   \else
3895     \ifx\@nnil#4\relax
3896       \edef\@glsccheckedmkidx{\the\@glstmpb\the\toks@}%
3897       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
3898         #4#5\@glxtr@endescspch}%
3899     \else
3900       \edef\@glsccheckedmkidx{\the\@glstmpb\the\toks@
3901         \@glxtr@autoindex@esc#1}%
3902       \def\@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
3903     \fi
3904   \fi
3905   \@glxtr@checkspch
3906 }
```

\Glossentrydesc Redefine to set the abbreviation format and accessibility support.

```
3907 \renewcommand*{\Glossentrydesc}[1]{%
3908   \glsoifexistsorwarn{#1}%
3909   {%
3910     \glsetabbrvfmt{\glscategory{#1}}%
3911     \Glsaccessdesc{#1}%
3912   }%
3913 }
```

lossentrysymbol Redefine to set the abbreviation format and accessibility support.

```
3914 \renewcommand*{\lossentrysymbol}[1]{%
3915   \glsoifexistsorwarn{#1}%
3916   {%
3917     \glsetabbrvfmt{\glscategory{#1}}%
3918     \Glsaccesssymbol{#1}%
3919   }%
3920 }
```

lossentrysymbol Redefine to set the abbreviation format and accessibility support.

```
3921 \renewcommand*{\Glossentrysymbol}[1]{%
3922   \glsdoifexistsorwarn{#1}%
3923   {%
3924     \glssetabbrvfmt{\glscategory{#1}}%
3925     \Glsaccesssymbol{#1}%
3926   }%
3927 }
```

Allow initials to be marked but only use the formatting for the tag in the glossary.

eInitialTagging Allow initial tagging. The first argument is a list of categories to apply this to. The second argument is the name of the command to use to tag the initials. This can't already be defined for safety unless the starred version is used.

```
3928 \newcommand*{\GlsXtrEnableInitialTagging}{%
3929   \@ifstar\s@glsextr@enabletagging\@glsextr@enabletagging
3930 }
3931 \@onlypreamble\GlsXtrEnableInitialTagging
```

r@enabletagging Starred version undefines command.

```
3932 \newcommand*{\s@glsextr@enabletagging}[2]{%
3933   \undef#2%
3934   \@glsextr@enabletagging{#1}{#2}%
3935 }
```

r@enabletagging Internal command.

```
3936 \newcommand*{\@glsextr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
3937   \@for\@glsextr@cat:=#1\do
3938   {%
3939     \ifdefempty\@glsextr@cat
3940     {}%
3941     {\glssetcategoryattribute{\@glsextr@cat}{tagging}{true}}%
3942   }%
3943   \newrobustcmd*#2[1]{##1}%
3944   \def\@glsextr@taggingcs{#2}%
3945   \renewcommand*\@glsextr@activate@initialtagging{%
3946     \let#2\@glsextr@tag
3947   }%
3948   \ifundef\@gls@preglossaryhook
3949   {\GlossariesExtraWarning{Initial tagging requires at least
3950     glossaries.sty v4.19 to work correctly}}%
3951   {}%
3952 }
```

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`

```
3953 \ifundef\mfu@checkword@do
3954 {
3955   \newcommand*{\mfu@checkword@do}[1]{%
3956     \ifdefstring{\mfu@checkword@arg}{#1}%
3957     {%
3958       \let\@mfu@domakefirstuc\@firstofone
3959       \listbreak
3960     }%
3961   }%
3962 }
```

`\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.

```
3963 \ifundef\mfu@checkword
3964 {
3965   \newcommand{\@glstr@do@titlecaps@warn}{%
3966     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
3967       support not available}%

```

One warning should suffice.

```
3968   \let\@glstr@do@titlecaps@warn\relax
3969 }
3970 }
3971 {
3972   \renewcommand*{\mfu@checkword}[1]{%
3973     \def\mfu@checkword@arg{#1}%
3974     \let\@mfu@domakefirstuc\makefirstuc
3975     \forlistloop\mfu@checkword@do\@mfu@nocaplist
3976   }
3977 }
3978 }
3979 {}% no patch required
```

`@titlecaps@warn` Do warning if title case not supported.

```
3980 \newcommand*{\@glstr@do@titlecaps@warn}{}
```

`@initialtagging` Used in `\printglossary` but at least v4.19 of `glossaries` required.

```
3981 \newcommand*{\@glstr@activate@initialtagging{}
```

`\@glstr@tag` Definition of tagging command when used in glossary.

```
3982 \newrobustcmd*{\@glstr@tag}[1]{%
3983   \gl@ifattribute{\gl@currententrylabel}{tagging}{true}%
3984   {\glstr@tagfont{#1}}{#1}%
3985 }
```

`\glstr@tagfont` Used in the glossary.

```
3986 \newcommand*{\glstr@tagfont}[1]{\underline{#1}}
```

`preglossaryhook` This macro was introduced in glossaries version 4.19, so it may not be defined. If it hasn't been defined this feature is unavailable. A check is added for the entry's existence to prevent errors from occurring if the user removes an entry or changes the label, which can interrupt the build process.

```

3987 \ifdef \@gls@preglossaryhook
3988 {
3989   \renewcommand*{\@gls@preglossaryhook}{%
3990     \@glsxtr@activate@initialtagging
3991     \let\@glsxtr@org@postdescription\glspostdescription
3992     \renewcommand*{\glspostdescription}{%
3993       \ifglsentryexists{\glscurrententrylabel}%
3994       {%
3995         \glsxtrpostdescription
3996         \@glsxtr@org@postdescription
3997       }{%
3998       }%

```

Enable the options used by `\@glsxtrp`:

```

3999   \glossxtrsetpopts
4000 }%
4001 }
4002 {}

```

`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.

```

4003 \newcommand*{\glsxtrpostdescription}{%
4004   \csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%
4005 }

```

`postdescgeneral`

```

4006 \newcommand*{\glsxtrpostdescgeneral}{}

```

`xtrpostdescterm`

```

4007 \newcommand*{\glsxtrpostdescterm}{}

```

`postdescacronym`

```

4008 \newcommand*{\glsxtrpostdescacronym}{}

```

`escabbreviation`

```

4009 \newcommand*{\glsxtrpostdescabbreviation}{}

```

`glspostlinkhook` Redefine the post link hook used by commands like `\gls` to make it easier for categories or attributes to modify this action. Since this hook occurs outside the existence check of commands like `\gls`, this needs to be checked again here. Do nothing if the entry hasn't been defined.

```

4010 \renewcommand*{\glspostlinkhook}{%
4011 \ifgl Sentryexists{\glslabel}{\glxtrpostlinkhook}{}%
4012 }

xtrpostlinkhook The entry label should already be stored in \glslabel by \@gls@link.
4013 \newcommand*{\glxtrpostlinkhook}{%
4014 \glxtrdiscardperiod{\glslabel}%
4015 {\glxtrpostlinkendsentence}%
4016 {\glxtrpostlink}%
4017 }

\glxtrpostlink
4018 \newcommand*{\glxtrpostlink}{%
4019 \csuse{glxtrpostlink\glscategory{\glslabel}}%
4020 }

linkendsentence Done by \glxtrpostlinkhook if a full stop is discarded.
4021 \newcommand*{\glxtrpostlinkendsentence}{%
4022 \ifcsdef{glxtrpostlink\glscategory{\glslabel}}
4023 {%
4024 \csuse{glxtrpostlink\glscategory{\glslabel}}%
Put the full stop back.
4025 .\spacefactor\sfcode'\. \relax
4026 }%
4027 {%
Assume the full stop was discarded because the entry ends with a period, so adjust the space-
factor.
4028 \spacefactor\sfcode'\. \relax
4029 }%
4030 }

dDescOnFirstUse Provide a command for appending the description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.
4031 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
4032 \glxtrifwasfirstuse{\space(\glssuccessdesc{\glslabel})}{}%
4033 }

ymbolOnFirstUse Provide a command for appending the symbol (if defined) in parentheses on first use, for the convenience of users wanting to add this to the post link hook.
4034 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
4035 \glxtrifwasfirstuse
4036 {%
4037 \ifglshassymbol{\glslabel}{\space(\glssuccesssymbol{\glslabel})}{}%
4038 }%
4039 {}%
4040 }

```

`trdiscardperiod` Discard following period (if present) if the `discardperiod` attribute is true. If a period is discarded, do the second argument otherwise do the third argument. The entry label is in the first argument. Since this is designed for abbreviations that end with a period, check if the plural form was used (which typically won't end with a period).

```

4041 \newcommand*\glstrdiscardperiod}[3]{%
4042   \glstrifwasfirstuse
4043   {%
4044     \gl@ifattribute{#1}{retainfirstuseperiod}{true}%
4045     {#3}%
4046     {%
4047       \gl@ifattribute{#1}{discardperiod}{true}%
4048       {%
4049         \gl@ifplural
4050         {%
4051           \gl@ifattribute{#1}{pluraldiscardperiod}{true}%
4052           {\glstrifperiod{#2}{#3}}%
4053           {#3}%
4054         }%
4055         {%
4056           \glstrifperiod{#2}{#3}%
4057         }%
4058       }%
4059     }%
4060   }%
4061 }%
4062 {%
4063   \gl@ifattribute{#1}{discardperiod}{true}%
4064   {%
4065     \gl@ifplural
4066     {%
4067       \gl@ifattribute{#1}{pluraldiscardperiod}{true}%
4068       {\glstrifperiod{#2}{#3}}%
4069       {#3}%
4070     }%
4071     {%
4072       \glstrifperiod{#2}{#3}%
4073     }%
4074   }%
4075   {#3}%
4076 }%
4077 }

```

`\glstrifperiod` 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`

```

4078 \newcommand*\glstrifperiod}[1]{\new@ifnextchar.\@firstoftwo{#1}}

```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`glstr@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 ' ').

```
4079 \newcommand*{\glxtr@punclist}{.,:;?!}
```

punctuationmark Add character to punctuation list.

```
4080 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}
```

punctuationmarks Reset the punctuation list.

```
4081 \newcommand*{\glxtrsetpunctuationmarks}[1]{\def\glxtr@punclist{#1}}
```

```
\glxtrifpunc \glxtrifnextpunc{<true part>}{<false part>}
```

Test if this is followed by a punctuation mark. (Adapted from \new@ifnextchar.)

```
4082 \newcommand*{\glxtrifnextpunc}[2]{%
4083   \def\reserved@a{#1}%
4084   \def\reserved@b{#2}%
4085   \futurelet\@glspunc@token\glxtr@ifnextpunc
4086 }
```

glxtr@ifnextpunc

```
4087 \newcommand*{\glxtr@ifnextpunc}{%
4088   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
4089   \reserved@b
4090 }
```

glxtr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
4091 \newcommand*{\glxtr@ifpunctoken}[1]{%
4092   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
4093 }
```

glxtr@ifpunctoken

```
4094 \def\@glxtr@ifpunctoken#1#2{%
4095   \let\reserved@d=#2%
4096   \ifx\reserved@d\@nnil
4097     \let\glxtr@next\@glxtr@notfoundinlist
4098   \else
4099     \ifx#1\reserved@d
4100       \let\glxtr@next\@glxtr@foundinlist
4101     \else
4102       \let\glxtr@next\@glxtr@ifpunctoken
4103     \fi
4104   \fi
4105   \glxtr@next#1%
4106 }
```

glxtr@foundinlist

```
4107 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
4108 \def\@glxstr@notfoundinlist#1{\@secondoftwo}
```

glxstrdopostpunc

```
\glxstrdopostpunc{<code>}
```

If this is followed by a punctuation character, do <code> after the character otherwise do <code> before whatever comes next.

```
4109 \newcommand{\glxstrdopostpunc}[1]{%
4110   \glxstrifnextpunc{\@glxstr@swaptwo{#1}}{#1}%
4111 }
```

@glxstr@swaptwo

```
4112 \newcommand{\@glxstr@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.

```
4113 \define@key{glxstrabbrv}{category}{%
4114   \edef\glscategorylabel{#1}%
4115   \ifcsdef{@glxstrabbrv@current@#1}%
4116   {%
```

Warning should already have been issued.

```
4117   \let\@glxstr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
4118   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
4119   \glxstr@applyabbrvstyle{\csname @glxstrabbrv@current@#1\endcsname}%
4120   \let\GlsXtrWarnDeprecatedAbbrStyle\@glxstr@orgwarndep
4121 }%
4122 {}%
4123 }
```

Save the short plural form. This may be needed before the entry is defined.

```
4124 \define@key{glxstrabbrv}{shortplural}{%
4125   \def\@glxstr@shortpl{#1}%
4126 }
```

Similarly for the long plural form.

```
4127 \define@key{glxstrabbrv}{longplural}{%
4128   \def\@glxstr@longpl{#1}%
4129 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

`\glsshortpltok`

```
4130 \newtoks\glsshortpltok
```

`\glslongpltok`

```
4131 \newtoks\glslongpltok
```

`sxtr@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.

```
4132 \newcommand*{\@glsxtr@insertdots}[2]{%
4133   \def#1{%
4134     \@glsxtr@insert@dots#1#2\@nnil
4135   }
```

`xtr@insert@dots`

```
4136 \newcommand*{\@glsxtr@insert@dots}[2]{%
4137   \ifx\@nnil#2\relax
4138   \let\@glsxtr@insert@dots@next\@gobble
4139   \else
4140     \ifx\relax#2\relax
4141     \else
4142       \appto#1{#2.}%
4143     \fi
4144     \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
4145     \fi
4146     \@glsxtr@insert@dots@next#1%
4147   }
```

`newabbreviation` Define a new generic abbreviation.

```
4148 \newcommand*{\newabbreviation}[4] [] {%
4149   \glskeylisttok{#1}%
4150   \glslabeltok{#2}%
4151   \glsshorttok{#3}%
4152   \glslongtok{#4}%
```

Get the category.

```
4153   \def\glscategorylabel{abbreviation}%
4154   \glsxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
4155   \setkeys*{glsxtrabbrv}[shortplural,longplural]{#1}%
```

Set the default long plural

```
4156   \def\@gls@longpl{#4\glspluralsuffix}%
```

Has the insertdots attribute been set?

```

4157 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
4158 {%
4159   \@glxtr@insertdots\@gls@short{#3}%
4160   \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
4161   \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
4162   {%
4163     \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
4164       '\abbrvpluralsuffix}%
4165   }%
4166   {%
4167     \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
4168     {%
4169       \let\@gls@shortpl\@gls@short
4170     }%
4171     {%
4172       \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
4173         \abbrvpluralsuffix}%
4174     }%
4175   }%
4176 }%
4177 {%

```

insertdots not true.

```

4178 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
4179 {%
4180   \def\@gls@shortpl{#3'\abbrvpluralsuffix}%
4181 }%
4182 {%
4183   \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
4184   {%
4185     \def\@gls@shortpl{#3}%
4186   }%
4187   {%
4188     \def\@gls@shortpl{#3\abbrvpluralsuffix}%
4189   }%
4190 }%
4191 }%

```

Hook for further customisation if required:

```

4192 \glxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%

```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary.

```

4193 \setkeys*{glxtrabbrv}[category]{#1}%

```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```

4194 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
4195 \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

4196 \newabbreviationhook
    Define this entry:
4197 \protected@edef\@do@newglossaryentry{%
4198 \noexpand\newglossaryentry{\the\glslabeltok}%
4199 {%
4200     type=\glstrabbrvtype,%
4201     category=abbreviation,%
4202     short={\the\glsshorttok},%
4203     shortplural={\the\glsshortpltok},%
4204     long={\the\glslongtok},%
4205     longplural={\the\glslongpltok},%
4206     name={\the\glsshorttok},%
4207     \CustomAbbreviationFields,%
4208     \the\glskeylisttok
4209     }%
4210 }%
4211 \@do@newglossaryentry
4212 \GlsXtrPostNewAbbreviation
4213 }

evpresetkeyhook Hook for extra stuff in \newabbreviation
4214 \newcommand*{\glstrnewabbspresetkeyhook}[3]{}

NewAbbreviation Hook used by abbreviation styles.
4215 \newcommand*{\GlsXtrPostNewAbbreviation}{}

bbreviationhook Hook for use with \newabbreviation.
4216 \newcommand*{\newabbreviationhook}{}

reviationFields
4217 \newcommand*{\CustomAbbreviationFields}{}

lsxtrfullformat Full format without case change.
4218 \newcommand*{\glsxtrfullformat}[2]{%
4219 \glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
4220 (\protect\glsfirstabbrvfont{\glsaccessshort{#1}})%
4221 }

lsxtrfullformat Full format with case change.
4222 \newcommand*{\Glsxtrfullformat}[2]{%
4223 \glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
4224 (\protect\glsfirstabbrvfont{\Glsaccessshort{#1}})%
4225 }

xtrfullplformat Plural full format without case change.
4226 \newcommand*{\glsxtrfullplformat}[2]{%
4227 \glsfirstlongfont{\glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
4228 (\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}})%
4229 }

```

`\glstrfullplformat` Plural full format with case change.

```
4230 \newcommand*{\Glsxtrfullplformat}[2]{%
4231   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glstrfullsep{#1}%
4232   (\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}})%
4233 }
```

`\glstrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
4234 \newcommand*{\glstrfullsep}[1]{\space}
```

In-line formats in case first use isn't compatible with `\glentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glstrinlinefullformat` Full format without case change.

```
4235 \newcommand*{\Glsxtrinlinefullformat}{\glstrfullformat}
```

`\glstrinlinefullformat` Full format with case change.

```
4236 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}
```

`\glstrfullplformat` Plural full format without case change.

```
4237 \newcommand*{\Glsxtrinlinefullplformat}{\glstrfullplformat}
```

`\glstrinlinefullplformat` Plural full format with case change.

```
4238 \newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}
```

Redefine `\glentryfull` etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the `\glstrfull` set of commands instead.

`\glentryfull`

```
4239 \renewcommand*{\glentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\Glsentryfull`

```
4240 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\glentryfullpl`

```
4241 \renewcommand*{\glentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\Glsentryfullpl`

```
4242 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
4243 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}
```

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.

```
4244 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}
```

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.

```
4245 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}
```

bbvrdefaultfont

```
4246 \newcommand*{\glsabbrvdefaultfont}[1]{#1}
```

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.

```
4247 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}
```

longdefaultfont Default font changing command used for the long form in commands like `\glsxtrlong`.

```
4248 \newcommand*{\glslongdefaultfont}[1]{#1}
```

lsfirstlongfont Font changing command used for the long form on first use or in the full format.

```
4249 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}
```

longdefaultfont

```
4250 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}
```

brvpluralsuffix Default plural suffix.

```
4251 \newcommand*{\abbrvpluralsuffix}{\glspluralsuffix}
```

`\glsxtrfull` Full form (no case-change).

```
4252 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\@ns@glsxtrfull}
```

```
4253 \newcommand*{\ns@glsxtrfull}[2][\%
```

```
4254 \new@ifnextchar[\@glsxtr@full{#1}{#2}]{%
```

```
4255 \@glsxtr@full{#1}{#2}[]}%
```

```
4256 }
```

`\@glsxtr@full` Low-level macro:

```
4257 \def\@glsxtr@full#1#2[#3]{%
```

```
4258 \glsdoifexists{#2}%
```

```
4259 {%
```

```
4260 \glssetabbrvfmt{\glscategory{#2}}%
```

```
4261 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
4262 \let\glsifplural\@secondoftwo
```

```
4263 \let\glsupscase\@firstofthree
```

```
4264 \let\glsinsert\@empty
```

```
4265 \def\glscustomtext{\glsxtrinlinefullformat{#2}{#3}}%
```

What should `\glsxtrifwasfirstuse` be set to here? Where the inline and display full forms are the same, this is essentially emulating first use, to it make sense for the postlink hook to pretend it was a first use instance. It makes less sense if the inline and display forms are different. Provide a hook to make it easier to reconfigure.

```
4266 \glsxtrsetupfulldefs
```

```
4267 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
```

```
4268 }%
```

```
4269 \glspostlinkhook
```

```
4270 }
```

trsetupfulldefs

```
4271 \newcommand*{\glxtrsetupfulldefs}{%
4272   \let\glxtrifwasfirstuse\@firstoftwo
4273 }
```

\Glsxtrfull Full form (first letter uppercase).

```
4274 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\@ns@Glsxtrfull}
4275 \newcommand*\ns@Glsxtrfull[2][{}]{%
4276   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}{%
4277     {\@Glsxtr@full{#1}{#2}[]}%
4278 }
```

\@Glsxtr@full Low-level macro:

```
4279 \def\@Glsxtr@full#1#2[#3]{%
4280   \glsdoifexists{#2}%
4281   {%
4282     \glssetabbrvfmt{\glscategory{#2}}%
4283     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4284     \let\glsifplural\@secondoftwo
4285     \let\glscapscase\@secondofthree
4286     \let\glsinsert\@empty
4287     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
4288     \glxtrsetupfulldefs
4289     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4290   }%
4291   \glspostlinkhook
4292 }
```

\GLSxtrfull Full form (all uppercase).

```
4293 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\@ns@GLSxtrfull}
4294 \newcommand*\ns@GLSxtrfull[2][{}]{%
4295   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}{%
4296     {\@GLSxtr@full{#1}{#2}[]}%
4297 }
```

\@GLSxtr@full Low-level macro:

```
4298 \def\@GLSxtr@full#1#2[#3]{%
4299   \glsdoifexists{#2}%
4300   {%
4301     \glssetabbrvfmt{\glscategory{#2}}%
4302     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4303     \let\glsifplural\@secondoftwo
4304     \let\glscapscase\@thirdofthree
4305     \let\glsinsert\@empty
4306     \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
4307     \glxtrsetupfulldefs
4308     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4309   }%
4310   \glspostlinkhook
```

4311 }

`\glxstrfullpl` Plural full form (no case-change).

```
4312 \newrobustcmd*{\glxstrfullpl}{\@gls@hyp@opt\ns@glxstrfullpl}
4313 \newcommand*\ns@glxstrfullpl[2][]{%
4314   \new@ifnextchar[{\@glxstr@fullpl{#1}{#2}}%
4315                   {\@glxstr@fullpl{#1}{#2}[]}%
4316 }
```

`\@glxstr@fullpl` Low-level macro:

```
4317 \def\@glxstr@fullpl#1#2[#3]{%
4318   \glsdoidexists{#2}%
4319   {%
4320     \glsssetabbrvfmt{\glscategory{#2}}%
4321     \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
4322     \let\gl@s@ifplural\@firstoftwo
4323     \let\gl@scapscase\@firstofthree
4324     \let\gl@sinsert\@empty
4325     \def\glscustomtext{\glxstrinlinefullplformat{#2}{#3}}%
4326     \glxstrsetupfulldefs
4327     \@gl@s@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
4328   }%
4329   \glspostlinkhook
4330 }
```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```
4331 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\ns@Glsxtrfullpl}
4332 \newcommand*\ns@Glsxtrfullpl[2][]{%
4333   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
4334                   {\@Glsxtr@fullpl{#1}{#2}[]}%
4335 }
```

`\@Glsxtr@fullpl` Low-level macro:

```
4336 \def\@Glsxtr@fullpl#1#2[#3]{%
4337   \glsdoidexists{#2}%
4338   {%
4339     \glsssetabbrvfmt{\glscategory{#2}}%
4340     \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
4341     \let\gl@s@ifplural\@firstoftwo
4342     \let\gl@scapscase\@secondofthree
4343     \let\gl@sinsert\@empty
4344     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
4345     \glxstrsetupfulldefs
4346     \@gl@s@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
4347   }%
4348   \glspostlinkhook
4349 }
```

`\GLSxtrfullpl` Plural full form (all upper case).

```

4350 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\ns@GLSxtrfullpl}
4351 \newcommand*\ns@GLSxtrfullpl[2] [] {%
4352   \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}]{%
4353     {\@GLSxtr@fullpl{#1}{#2} []}%
4354 }

```

\@GLSxtr@fullpl Low-level macro:

```

4355 \def\@GLSxtr@fullpl#1#2[#3] {%
4356   \glsdoifexists{#2}%
4357   {%
4358     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4359     \let\glsifplural\@firstoftwo
4360     \let\glscapscase\@thirdofthree
4361     \let\glsinsert\@empty
4362     \def\glscustomtext{%
4363       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
4364     \glsxtrsetupfulldefs
4365     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4366   }%
4367   \glspostlinkhook
4368 }

```

The short and long forms work in a similar way to acronyms.

\glsxtrshort

```

4369 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4370 \newcommand*{\ns@glsxtrshort}[2] [] {%
4371   \new@ifnextchar[{\@glsxtrshort{#1}{#2}}]{\@glsxtrshort{#1}{#2} []}%
4372 }

```

Read in the final optional argument:

```

4373 \def\@glsxtrshort#1#2[#3] {%
4374   \glsdoifexists{#2}%
4375   {%

```

Need to make sure \glsabbrvfont is set correctly.

```

4376   \glssetabbrvfmt{\glscategory{#2}}%
4377   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4378   \let\glsxtrifwasfirstuse\@secondoftwo
4379   \let\glsifplural\@secondoftwo
4380   \let\glscapscase\@firstofthree
4381   \let\glsinsert\@empty
4382   \def\glscustomtext{%
4383     \glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
4384     \ifglsxtrininsertinside\else#3\fi
4385   }%
4386   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4387   }%
4388   \glspostlinkhook

```

4389 }

\Glsxtrshort

4390 \newrobustcmd*{\Glsxtrshort}{\@gls@hyp@opt\@ns@Glsxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

4391 \newcommand*{\ns@Glsxtrshort}[2] [] {%

4392 \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} [] }%

4393 }

Read in the final optional argument:

4394 \def\@Glsxtrshort#1#2[#3] {%

4395 \glsdoifexists{#2}%

4396 {%

4397 \glssetabbrvfmt{\glscategory{#2}}%

4398 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

4399 \let\glsxtrifwasfirstuse\@secondoftwo

4400 \let\glsifplural\@secondoftwo

4401 \let\glsapscase\@secondofthree

4402 \let\glsinsert\@empty

4403 \def\glscustomtext{%

4404 \glsabbrvfont{\Glsaccessshort{#2}}\ifglsxtrininsertinside#3\fi}%

4405 \ifglsxtrininsertinside\else#3\fi

4406 }%

4407 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%

4408 }%

4409 \glspostlinkhook

4410 }

\GLSxtrshort

4411 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

4412 \newcommand*{\ns@GLSxtrshort}[2] [] {%

4413 \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} [] }%

4414 }

Read in the final optional argument:

4415 \def\@GLSxtrshort#1#2[#3] {%

4416 \glsdoifexists{#2}%

4417 {%

4418 \glssetabbrvfmt{\glscategory{#2}}%

4419 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

4420 \let\glsxtrifwasfirstuse\@secondoftwo

4421 \let\glsifplural\@secondoftwo

4422 \let\glsapscase\@thirdofthree

4423 \let\glsinsert\@empty

4424 \def\glscustomtext{%

4425 \mfirstucMakeUppercase

4426 {\glsabbrvfont{\Glsaccessshort{#2}}\ifglsxtrininsertinside#3\fi}%

4427 \ifglsxtrininsertinside\else#3\fi

```

4428     }%
4429     }%
4430     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4431     }%
4432     \glspostlinkhook
4433 }

```

\glsxtrlong

```

4434 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
4435 \newcommand*{\ns@glsxtrlong}[2][{}]{%
4436   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2}[]}%
4437 }

    Read in the final optional argument:
4438 \def\@glsxtrlong#1#2[#3]{%
4439   \glsdoifexists{#2}%
4440   {%
4441     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4442     \let\glsxtrifwasfirstuse\@secondoftwo
4443     \let\glsifplural\@secondoftwo
4444     \let\glscapscase\@firstofthree
4445     \let\glsinsert\@empty
4446     \def\glscustomtext{%
4447       \glsfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
4448       \ifglsxtrininsertinside\else#3\fi
4449     }%
4450     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4451     }%
4452     \glspostlinkhook
4453 }

```

\Glsxtrlong

```

4454 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
4455 \newcommand*{\ns@Glsxtrlong}[2][{}]{%
4456   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2}[]}%
4457 }

    Read in the final optional argument:
4458 \def\@Glsxtrlong#1#2[#3]{%
4459   \glsdoifexists{#2}%
4460   {%
4461     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4462     \let\glsxtrifwasfirstuse\@secondoftwo
4463     \let\glsifplural\@secondoftwo
4464     \let\glscapscase\@secondofthree
4465     \let\glsinsert\@empty
4466     \def\glscustomtext{%

```

```

4467     \glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
4468     \ifglxtrinsertinside\else#3\fi
4469   }%
4470   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4471 }%
4472 \glspostlinkhook
4473 }

```

\GLSxtrlong

```

4474 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}

  Define the un-starred form. Need to determine if there is a final optional argument
4475 \newcommand*{\ns@GLSxtrlong}[2] [] {%
4476   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
4477 }

  Read in the final optional argument:
4478 \def\@GLSxtrlong#1#2[#3] {%
4479   \glstoifexists{#2}%
4480   {%
4481     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4482     \let\glxtrifwasfirstuse\@secondoftwo
4483     \let\glsifplural\@secondoftwo
4484     \let\glscapscase\@thirdofthree
4485     \let\glsinsert\@empty
4486     \def\glscustomtext{%
4487       \mfirstucMakeUppercase
4488       {\glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
4489       \ifglxtrinsertinside\else#3\fi
4490     }%
4491   }%
4492   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4493 }%
4494 \glspostlinkhook
4495 }

```

Plural short forms:

\glxtrshortpl

```

4496 \newrobustcmd*{\glxtrshortpl}{\@gls@hyp@opt\ns@glxtrshortpl}

  Define the un-starred form. Need to determine if there is a final optional argument
4497 \newcommand*{\ns@glxtrshortpl}[2] [] {%
4498   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2} []}%
4499 }

  Read in the final optional argument:
4500 \def\@glxtrshortpl#1#2[#3] {%
4501   \glstoifexists{#2}%
4502   {%
4503     \glsssetabbrvfmt{\glscategory{#2}}%

```

```

4504 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4505 \let\glxtrifwasfirstuse\@secondoftwo
4506 \let\gl@sifplural\@firstoftwo
4507 \let\gl@scapscase\@firstofthree
4508 \let\gl@insert\@empty
4509 \def\glscustomtext{%
4510     \gl@sabbrvfont{\gl@accessshortpl{#2}\ifglxtrininsertinside#3\fi}%
4511     \ifglxtrininsertinside\else#3\fi
4512 }%
4513 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4514 }%
4515 \glspostlinkhook
4516 }

```

\Glsxtrshortpl

```

4517 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\ns@Glsxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
4518 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
4519     \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
4520 }

    Read in the final optional argument:
4521 \def\@Glsxtrshortpl#1#2[#3] {%
4522     \gl@doifexists{#2}%
4523     {%
4524         \gl@setabbrvfmt{\gl@category{#2}}%
4525         \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4526         \let\glxtrifwasfirstuse\@secondoftwo
4527         \let\gl@sifplural\@firstoftwo
4528         \let\gl@scapscase\@secondofthree
4529         \let\gl@insert\@empty
4530         \def\glscustomtext{%
4531             \gl@sabbrvfont{\Glsaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
4532             \ifglxtrininsertinside\else#3\fi
4533         }%
4534         \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4535     }%
4536     \glspostlinkhook
4537 }

```

\GLSxtrshortpl

```

4538 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
4539 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
4540     \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
4541 }

    Read in the final optional argument:
4542 \def\@GLSxtrshortpl#1#2[#3] {%

```

```

4543 \glsdoifexists{#2}%
4544 {%
4545   \glssetabbrvfmt{\glscategory{#2}}%
4546   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4547   \let\glsxtrifwasfirstuse\@secondoftwo
4548   \let\glsifplural\@firstoftwo
4549   \let\glsapscase\@thirdofthree
4550   \let\glsinsert\@empty
4551   \def\glscustomtext{%
4552     \mfirstucMakeUppercase
4553     {\glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
4554     \ifglsxtrininsertinside\else#3\fi
4555   }%
4556   }%
4557   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4558 }%
4559 \glspostlinkhook
4560 }

```

Plural long forms:

\glsxtrlongpl

```

4561 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\@ns@glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4562 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
4563   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
4564 }

```

Read in the final optional argument:

```

4565 \def\@glsxtrlongpl#1#2[#3]{%
4566   \glsdoifexists{#2}%
4567   {%
4568     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4569     \let\glsxtrifwasfirstuse\@secondoftwo
4570     \let\glsifplural\@firstoftwo
4571     \let\glsapscase\@firstofthree
4572     \let\glsinsert\@empty
4573     \def\glscustomtext{%
4574       \glsfont{\glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
4575       \ifglsxtrininsertinside\else#3\fi
4576     }%
4577     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4578   }%
4579   \glspostlinkhook
4580 }

```

\Glsxtrlongpl

```

4581 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns\Glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```
4582 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
4583   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
4584 }
```

Read in the final optional argument:

```
4585 \def\@Glsxtrlongpl#1#2[#3]{%
4586   \glsdoifexists{#2}%
4587   {%
4588     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4589     \let\glxtrifwasfirstuse\@secondoftwo
4590     \let\glsifplural\@firstoftwo
4591     \let\glscapscase\@secondofthree
4592     \let\glsinsert\@empty
4593     \def\glscustomtext{%
4594       \glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
4595       \ifglxtrininsertinside\else#3\fi
4596     }%
4597     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4598   }%
4599   \glspostlinkhook
4600 }
```

\GLSxtrlongpl

```
4601 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\ns@GLSxtrlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
4602 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
4603   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
4604 }
```

Read in the final optional argument:

```
4605 \def\@GLSxtrlongpl#1#2[#3]{%
4606   \glsdoifexists{#2}%
4607   {%
4608     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4609     \let\glxtrifwasfirstuse\@secondoftwo
4610     \let\glsifplural\@firstoftwo
4611     \let\glscapscase\@thirdofthree
4612     \let\glsinsert\@empty
4613     \def\glscustomtext{%
4614       \mfirstucMakeUppercase
4615       {\glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
4616       \ifglxtrininsertinside\else#3\fi
4617     }%
4618   }%
4619   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4620 }%
4621 \glspostlinkhook
4622 }
```

`\glssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```
4623 \newcommand*{\glssetabbrvfmt}[1]{%
4624   \ifcsdef{@glsabbrv@current@#1}%
4625     {\glsxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
4626     {\glsxtr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
4627 }
```

`sxtrngenabbrvfmt` Similar to `\glsngenacfmt`, but for abbreviations.

```
4628 \newcommand*{\glsxtrngenabbrvfmt}{%
4629   \ifdefempty\glscustomtext
4630     {%
4631       \ifglsused\glslabel
4632       {%
```

Subsequent use:

```
4633       \glsifplural
4634       {%
```

Subsequent plural form:

```
4635       \glscapscase
4636       {%
```

Subsequent plural form, don't adjust case:

```
4637       \glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert
4638       }%
4639       {%
```

Subsequent plural form, make first letter upper case:

```
4640       \glsabbrvfont{\Glsaccessshortpl{\glslabel}}\glsinsert
4641       }%
4642       {%
```

Subsequent plural form, all caps:

```
4643       \mfirstucMakeUppercase
4644       {\glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert}%
4645       }%
4646       }%
4647       {%
```

Subsequent singular form

```
4648       \glscapscase
4649       {%
```

Subsequent singular form, don't adjust case:

```
4650       \glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert
4651       }%
4652       {%
```

Subsequent singular form, make first letter upper case:

```
4653       \glsabbrvfont{\Glsaccessshort{\glslabel}}\glsinsert
4654       }%
4655       {%
```

Subsequent singular form, all caps:

```
4656      \mfirstucMakeUppercase
4657      {\glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert}%
4658      }%
4659      }%
4660      }%
4661      {%
```

First use:

```
4662      \glsifplural
4663      {%
```

First use plural form:

```
4664      \glscapscase
4665      {%
```

First use plural form, don't adjust case:

```
4666      \glxtrfullplformat{\glslabel}{\glsinsert}%
4667      }%
4668      {%
```

First use plural form, make first letter upper case:

```
4669      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
4670      }%
4671      {%
```

First use plural form, all caps:

```
4672      \mfirstucMakeUppercase
4673      {\glxtrfullplformat{\glslabel}{\glsinsert}}%
4674      }%
4675      }%
4676      {%
```

First use singular form

```
4677      \glscapscase
4678      {%
```

First use singular form, don't adjust case:

```
4679      \glxtrfullformat{\glslabel}{\glsinsert}%
4680      }%
4681      {%
```

First use singular form, make first letter upper case:

```
4682      \Glsxtrfullformat{\glslabel}{\glsinsert}%
4683      }%
4684      {%
```

First use singular form, all caps:

```
4685      \mfirstucMakeUppercase
4686      {\glxtrfullformat{\glslabel}{\glsinsert}}%
4687      }%
4688      }%
4689      }%
```

```

4690 }%
4691 {%
    User supplied text.
4692     \glscustomtext
4693 }%
4694 }

```

1.6.1 Abbreviation Styles Setup

abbreviationstyle

```

4695 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
4696     \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
4697     {%
4698         \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
4699     }%
4700     {%

```

Have abbreviations already been defined for this category?

```

4701     \ifcsstring{@glsabbrv@current@#1}{#2}%
4702     {%

```

Style already set.

```

4703     }%
4704     {%
4705         \def\@glsxtr@dostylewarn{%
4706             \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
4707             {%
4708                 \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
4709                     style has been switched \MessageBreak
4710                     for category ‘#1’, \MessageBreak
4711                     but there have already been entries \MessageBreak
4712                     defined for this category. Unwanted \MessageBreak
4713                     side-effects may result}}%
4714                 \@endfortrue
4715             }%
4716             \@glsxtr@dostylewarn

```

Set up the style for the given category.

```

4717         \csdef{@glsabbrv@current@#1}{#2}%
4718         \glsxtr@applyabbrvstyle{#2}%
4719     }%
4720 }%
4721 }

```

applyabbrvstyle Apply the abbreviation style without existence check.

```

4722 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
4723     \csuse{@glsabbrv@dispstyle@setup@#1}%
4724     \csuse{@glsabbrv@dispstyle@fmts@#1}%
4725 }

```

r@applyabbrvfmt Only apply the style formats.

```
4726 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
4727   \csuse{@glsabbrv@dispstyle@fmts@#1}%
4728 }
```

breivationstyle This is different from \newacronymstyle. The first argument is the label, the second argument sets the information required when defining the new abbreviation and the third argument sets the commands used to display the full format.

```
4729 \newcommand*{\newabbreviationstyle}[3]{%
4730   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
4731   {%
4732     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
4733       defined}{}%
4734   }%
4735   {%
4736     \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
4737     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
4738     #2}%
4739     \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
4740     \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
4741     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
4742     \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
4743     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
4744     #3}%
4745   }%
4746 }
```

breivationstyle

```
4747 \newcommand*{\renewabbreviationstyle}[3]{%
4748   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
4749   {%
4750     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
4751   }%
4752   {%
4753     \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
4754     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
4755     #2}%
4756     \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
4757     \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
4758     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
4759     \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
4760     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

```

4761     #3}%
4762   }%
4763 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

4764 \newcommand*{\letabbreviationstyle}[2]{%
4765   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
4766   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
4767 }

```

deprecated@abbrstyle `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

4768 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
4769   \csdef{@glsabbrv@dispstyle@setup@#1}{%
4770     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
4771     \csuse{@glsabbrv@dispstyle@setup@#2}%
4772   }%
4773   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
4774 }

```

deprecatedAbbrStyle Generate warning for deprecated style use.

```

4775 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
4776   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
4777   use ‘#2’ instead}%
4778 }

```

useAbbrStyleSetup

```

4779 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
4780   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
4781   {%
4782     \PackageError{glossaries-extra}%
4783     {Unknown abbreviation style definitions ‘#1’}{}%
4784   }%
4785   {%
4786     \csname @glsabbrv@dispstyle@setup@#1\endcsname
4787   }%
4788 }

```

useAbbrStyleFmt

```

4789 \newcommand*{\GlsXtrUseAbbrStyleFmt}[1]{%
4790   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
4791   {%
4792     \PackageError{glossaries-extra}%
4793     {Unknown abbreviation style formats ‘#1’}{}%

```

```

4794 }%
4795 {%
4796   \csname @glsabbrv@dispstyle@fmts@#1\endcsname
4797 }%
4798 }

```

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.

```

4799 \newif\ifglsxtrinsertinside
4800 \glsxtrinsertinsidefalse

```

`long-short`

```

4801 \newabbreviationstyle{long-short}%
4802 {%
4803   \renewcommand*{\CustomAbbreviationFields}{%
4804     name={\protect\glsabbrvfont{\the\glsshorttok}},
4805     sort={\the\glsshorttok},
4806     first={\protect\glsfirstlongfont{\the\glslongtok}}%
4807     \protect\glsxtrfullsep{\the\glslabeltok}}%
4808     (\protect\glsfirstabbrvfont{\the\glsshorttok}}),%
4809     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%
4810     \protect\glsxtrfullsep{\the\glslabeltok}}%
4811     (\protect\glsfirstabbrvfont{\the\glsshortpltok}}),%
4812     plural={\protect\glsabbvfont{\the\glsshortpltok}}},%
4813     description={\the\glslongtok}}%

```

Unset the regular attribute if it has been set.

```

4814 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4815   \glshasattribute{\the\glslabeltok}{regular}%
4816   {%
4817     \glssetattribute{\the\glslabeltok}{regular}{false}%
4818   }%
4819   {}%
4820 }%
4821 }%
4822 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4823 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%

```

```

4824 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4825 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4826 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4827 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

4828 \renewcommand*{\glsxtrfullformat}[2]{%
4829   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
4830   \ifglsxtrininsertinside\else##2\fi
4831   \glsxtrfullsep{##1}%
4832   (\glsfirstabbrvfont{\glsaccessshort{##1}})%
4833 }%
4834 \renewcommand*{\glsxtrfullplformat}[2]{%
4835   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
4836   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4837   (\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4838 }%
4839 \renewcommand*{\Glsxtrfullformat}[2]{%
4840   \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
4841   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4842   (\glsfirstabbrvfont{\Glsaccessshort{##1}})%
4843 }%
4844 \renewcommand*{\Glsxtrfullplformat}[2]{%
4845   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
4846   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4847   (\glsfirstabbrvfont{\Glsaccessshortpl{##1}})%
4848 }%
4849 }

```

Set this as the default style for general abbreviations:

```

4850 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

4851 \newcommand*{\glsxtrlongshortdescsort}{\the\glslongtok\space(\the\glsshorttok)}

```

long-short-desc User supplies description. The long form is included in the name.

```

4852 \newabbreviationstyle{long-short-desc}%
4853 {%
4854   \renewcommand*{\CustomAbbreviationFields}{%
4855     name={\protect\glsxtrfullformat{\the\glslabeltok}{}},
4856     sort={\glsxtrlongshortdescsort},%
4857     first={\protect\glsfirstlongfont{\the\glslongtok}%
4858       \protect\glsxtrfullsep{\the\glslabeltok}%
4859       (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
4860     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
4861       \protect\glsxtrfullsep{\the\glslabeltok}%
4862       (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%

```

The text key should only have the short form.

```

4863   text={\protect\glsabbrvfont{\the\glsshorttok}},%

```

```

4864 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
4865 }%

Unset the regular attribute if it has been set.
4866 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4867   \glshasattribute{\the\glslabeltok}{regular}%
4868   {%
4869     \glissetattribute{\the\glslabeltok}{regular}{false}%
4870   }%
4871   {}%
4872 }%
4873 }%
4874 {%
4875   \GlsXtrUseAbbrStyleFmts{long-short}%
4876 }

```

short-long Short form followed by long form in parenthesis on first use.

```

4877 \newabbreviationstyle{short-long}%
4878 {%
4879   \renewcommand*{\CustomAbbreviationFields}{%
4880     name={\protect\glsabbrvfont{\the\glsshortttok}},
4881     sort={\the\glsshortttok},
4882     description={\the\glslongtok},%
4883     first={\protect\glsfirstabbrvfont{\the\glsshortttok}}%
4884     \protect\glsxtrfullsep{\the\glslabeltok}%
4885     (\protect\glsfirstlongfont{\the\glslongtok})),%
4886     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
4887     \protect\glsxtrfullsep{\the\glslabeltok}%
4888     (\protect\glsfirstlongfont{\the\glslongpltok})),%
4889     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
4890   }%
4891   \glshasattribute{\the\glslabeltok}{regular}%
4892   {%
4893     \glissetattribute{\the\glslabeltok}{regular}{false}%
4894   }%
4895   {}%
4896 }%
4897 }%
4898 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4899 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4900 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4901 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4902 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4903 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

4904 \renewcommand*{\glsxtrfullformat}[2]{%

```

```

4905 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4906 \ifglxtrinsertinside\else##2\fi
4907 \glxtrfullsep{##1}%
4908 (\glsfirstlongfont{\glsaccesslong{##1}})%
4909 }%
4910 \renewcommand*{\glxtrfullplformat}[2]{%
4911 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4912 \ifglxtrinsertinside\else##2\fi
4913 \glxtrfullsep{##1}%
4914 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4915 }%
4916 \renewcommand*{\Glsxtrfullformat}[2]{%
4917 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4918 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4919 (\glsfirstlongfont{\glsaccesslong{##1}})%
4920 }%
4921 \renewcommand*{\Glsxtrfullplformat}[2]{%
4922 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4923 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4924 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4925 }%
4926 }

```

short-long-desc User supplies description. The long form is included in the name.

```

4927 \newabbreviationstyle{short-long-desc}%
4928 {%
4929 \renewcommand*{\CustomAbbreviationFields}{%
4930 name={\protect\glxtrfullformat{\the\glslabeltok}{}},
4931 sort={\the\glsshorttok},%
4932 first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
4933 \protect\glxtrfullsep{\the\glslabeltok}%
4934 (\protect\glsfirstlongfont{\the\glslongtok}}),%
4935 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
4936 \protect\glxtrfullsep{\the\glslabeltok}%
4937 (\protect\glsfirstlongfont{\the\glslongpltok}}),%
4938 text={\protect\glsabbrvfont{\the\glsshorttok}},%
4939 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
4940 }%

```

Unset the regular attribute if it has been set.

```

4941 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4942 \glshasattribute{\the\glslabeltok}{regular}%
4943 {%
4944 \glssetattribute{\the\glslabeltok}{regular}{false}%
4945 }%
4946 {}%
4947 }%
4948 }%

```

```

4949 {%
4950   \GlsXtrUseAbbrStyleFmts{short-long}%
4951 }

```

ongfootnotefont Only used by the “footnote” styles.

```

4952 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%

```

ongfootnotefont Only used by the “footnote” styles.

```

4953 \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`).

```

4954 \newcommand*{\glsxtrabbrvfootnote}[2]{\footnote{#2}}

```

footnote Short form followed by long form in footnote on first use.

```

4955 \newabbreviationstyle{footnote}%
4956 {%
4957   \renewcommand*{\CustomAbbreviationFields}{%
4958     name={\protect\glsabbrvfont{\the\glsshorttok}},
4959     sort={\the\glsshorttok},
4960     description={\the\glslongtok},%

4961     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
4962     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
4963     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
4964     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
4965     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
4966     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
4967     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

4968   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4969     \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
4970     \glsattribute{\the\glslabeltok}{regular}%
4971     {%
4972       \glssetattribute{\the\glslabeltok}{regular}{false}%
4973     }%
4974   }%
4975 }%
4976 }%
4977 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4978 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4979 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4980 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4981 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
4982 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

4983 \renewcommand*{\glsxtrfullformat}[2]{%
4984   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4985   \ifglsxtrininsertinside\else##2\fi
4986   \protect\glsxtrabbrvfootnote{##1}%
4987   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
4988 }%
4989 \renewcommand*{\glsxtrfullplformat}[2]{%
4990   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4991   \ifglsxtrininsertinside\else##2\fi
4992   \protect\glsxtrabbrvfootnote{##1}%
4993   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
4994 }%
4995 \renewcommand*{\Glsxtrfullformat}[2]{%
4996   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4997   \ifglsxtrininsertinside\else##2\fi
4998   \protect\glsxtrabbrvfootnote{##1}%
4999   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
5000 }%
5001 \renewcommand*{\Glsxtrfullplformat}[2]{%
5002   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5003   \ifglsxtrininsertinside\else##2\fi
5004   \protect\glsxtrabbrvfootnote{##1}%
5005   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
5006 }%

```

The first use full form and the inline full form use the short (long) style.

```

5007 \renewcommand*{\glsxtrinlinefullformat}[2]{%
5008   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5009   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5010   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
5011 }%
5012 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
5013   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5014   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5015   (\glsfirstlongfootnotefont{\glsaccesslongpl{##1}})%
5016 }%
5017 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
5018   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5019   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5020   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
5021 }%
5022 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%

```

```

5023 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5024 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5025 (\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}})%
5026 }%
5027 }

```

short-footnote

```

5028 \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.

```

5029 \newabbreviationstyle{postfootnote}%
5030 {%
5031 \renewcommand*{\CustomAbbreviationFields}{%
5032 name={\protect\glsabbrvfont{\the\glsshorttok}},
5033 sort={\the\glsshorttok},
5034 description={\the\glslongtok},%
5035 first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
5036 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
5037 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

5038 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5039 \csdef{glxtrpostlink\glscategorylabel}{%
5040 \glxtrifwasfirstuse
5041 }%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

5042 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
5043 {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
5044 }%
5045 {}%
5046 }%
5047 \glshasattribute{\the\glslabeltok}{regular}%
5048 {%
5049 \glissetattribute{\the\glslabeltok}{regular}{false}%
5050 }%
5051 {}%
5052 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

5053 \renewcommand*{\glxtrsetupfulldefs}{%
5054 \let\glxtrifwasfirstuse\@secondoftwo
5055 }%
5056 }%
5057 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5058 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
5059 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5060 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5061 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
5062 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

5063 \renewcommand*{\glsxtrfullformat}[2]{%
5064   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5065   \ifglsxtrininsertinside\else##2\fi
5066 }%
5067 \renewcommand*{\glsxtrfullplformat}[2]{%
5068   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5069   \ifglsxtrininsertinside\else##2\fi
5070 }%
5071 \renewcommand*{\Glsxtrfullformat}[2]{%
5072   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5073   \ifglsxtrininsertinside\else##2\fi
5074 }%
5075 \renewcommand*{\Glsxtrfullplformat}[2]{%
5076   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5077   \ifglsxtrininsertinside\else##2\fi
5078 }%

```

The first use full form and the inline full form use the short (long) style.

```

5079 \renewcommand*{\glsxtrinlinefullformat}[2]{%
5080   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5081   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5082   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
5083 }%
5084 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
5085   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5086   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5087   (\glsfirstlongfootnotefont{\glsaccesslongpl{##1}})%
5088 }%
5089 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
5090   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5091   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5092   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
5093 }%
5094 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
5095   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5096   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5097   (\glsfirstlongfootnotefont{\glsaccesslongpl{##1}})%
5098 }%
5099 }

```

rt-postfootnote

```

5100 \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.

```

5101 \newabbreviationstyle{short}%
5102 {%
5103   \renewcommand*{\CustomAbbreviationFields}{%
5104     name={\protect\glsabbrvfont{\the\glsshorttok}},
5105     sort={\the\glsshorttok},
5106     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
5107     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
5108     text={\protect\glsabbrvfont{\the\glsshorttok}},
5109     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
5110     description={\the\glslongtok}}%
5111   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5112     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5113 }%
5114 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5115 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
5116 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5117 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5118 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5119 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

5120 \renewcommand*{\glsxtrinlinefullformat}[2]{%
5121   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
5122   \ifglsxtrininsertinside##2\fi}%
5123   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5124   (\glsfirstlongfont{\glsaccesslong{##1}})%
5125 }%
5126 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
5127   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
5128   \ifglsxtrininsertinside##2\fi}%
5129   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5130   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
5131 }%
5132 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
5133   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
5134   \ifglsxtrininsertinside##2\fi}%
5135   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5136   (\glsfirstlongfont{\Glsaccesslong{##1}})%
5137 }%
5138 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
5139   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
5140   \ifglsxtrininsertinside##2\fi}%
5141   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5142   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%

```

5143 }%

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.

```

5144 \renewcommand*{\glxtrfullformat}[2]{%
5145   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5146   \ifglxtrinsertinside\else##2\fi
5147 }%
5148 \renewcommand*{\glxtrfullplformat}[2]{%
5149   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5150   \ifglxtrinsertinside\else##2\fi
5151 }%
5152 \renewcommand*{\Glsxtrfullformat}[2]{%
5153   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5154   \ifglxtrinsertinside\else##2\fi
5155 }%
5156 \renewcommand*{\Glsxtrfullplformat}[2]{%
5157   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5158   \ifglxtrinsertinside\else##2\fi
5159 }%
5160 }

```

Set this as the default style for acronyms:

```
5161 \setabbreviationstyle[acronym]{short}
```

short-nolong

```
5162 \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.

```

5163 \newabbreviationstyle{short-desc}%
5164 {%
5165   \renewcommand*{\CustomAbbreviationFields}{%
5166     name={\protect\glxtrinlinefullformat{\the\glslabeltok}{}},
5167     sort={\the\glsshorttok},
5168     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
5169     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
5170     text={\protect\glsabbrvfont{\the\glsshorttok}},
5171     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
5172     description={\the\glslongtok}}%
5173   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5174     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5175 }%
5176 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5177 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
5178 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5179 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%

```

```

5180 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5181 \renewcommand*\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

5182 \renewcommand*\glsxtrinlinefullformat}[2]{%
5183   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5184   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5185   (\glsfirstlongfont{\glsaccesslong{##1}})%
5186 }%
5187 \renewcommand*\glsxtrinlinefullplformat}[2]{%
5188   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5189   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5190   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
5191 }%
5192 \renewcommand*\Glsxtrinlinefullformat}[2]{%
5193   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5194   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5195   (\glsfirstlongfont{\Glsaccesslong{##1}})%
5196 }%
5197 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
5198   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5199   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5200   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
5201 }%

```

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.

```

5202 \renewcommand*\glsxtrfullformat}[2]{%
5203   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5204   \ifglsxtrininsertinside\else##2\fi
5205 }%
5206 \renewcommand*\glsxtrfullplformat}[2]{%
5207   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5208   \ifglsxtrininsertinside\else##2\fi
5209 }%
5210 \renewcommand*\Glsxtrfullformat}[2]{%
5211   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5212   \ifglsxtrininsertinside\else##2\fi
5213 }%
5214 \renewcommand*\Glsxtrfullplformat}[2]{%
5215   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5216   \ifglsxtrininsertinside\else##2\fi
5217 }%
5218 }

```

ort-nolong-desc

```

5219 \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.

```

5220 \newabbreviationstyle{long-desc}%
5221 {%
5222   \renewcommand*{\CustomAbbreviationFields}{%
5223     name={\protect\protect\glsfirstlongfont{\the\glslongtok}},
5224     sort={\the\glslongtok},
5225     first={\protect\glsfirstlongfont{\the\glslongtok}},
5226     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5227     text={\the\glslongtok},
5228     plural={\the\glslongpltok}%
5229   }%
5230   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5231     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5232 }%
5233 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5234 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
5235 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5236 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5237 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5238 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the long format followed by the short form in parentheses.

```

5239 \renewcommand*{\glsxtrinlinefullformat}[2]{%
5240   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5241   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5242   (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
5243 }%
5244 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
5245   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
5246   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5247   (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
5248 }%
5249 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
5250   \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5251   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5252   (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
5253 }%
5254 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
5255   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
5256   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5257   (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
5258 }%

```

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.

```

5259 \renewcommand*{\glsxtrfullformat}[2]{%
5260   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5261   \ifglsxtrininsertinside\else##2\fi

```

```

5262 }%
5263 \renewcommand*{\glxtrfullplformat}[2]{%
5264   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5265   \ifglxtrinsertinside\else##2\fi
5266 }%
5267 \renewcommand*{\Glsxtrfullformat}[2]{%
5268   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
5269   \ifglxtrinsertinside\else##2\fi
5270 }%
5271 \renewcommand*{\Glsxtrfullplformat}[2]{%
5272   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5273   \ifglxtrinsertinside\else##2\fi
5274 }%
5275 }

```

ng-noshort-desc Provide a synonym that matches similar styles.

```
5276 \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.

```

5277 \newabbreviationstyle{long}%
5278 {%
5279   \renewcommand*{\CustomAbbreviationFields}{%
5280     name={\protect\glsabbrvfont{\the\glsshorttok}},
5281     sort={\the\glsshorttok},
5282     first={\protect\glsfirstlongfont{\the\glslongtok}},
5283     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5284     text={\the\glslongtok},
5285     plural={\the\glslongpltok},%
5286     description={\the\glslongtok}%
5287   }%
5288   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5289     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5290 }%
5291 {%
5292   \GlsXtrUseAbbrStyleFmts{long-desc}%
5293 }

```

long-noshort Provide a synonym that matches similar styles.

```
5294 \letabbreviationstyle{long-noshort}{long}
```

1.6.3 Predefined Styles (Small Capitals)

These styles use:

\glxtrscfont

```
5295 \newcommand*{\glxtrscfont}[1]{\textsc{#1}}
```

sxtrfirstscfont

```
5296 \newcommand*{\glxtrfirstscfont}[1]{\glxtrscfont{#1}}
```

and for the default short form suffix:

\glxtrscsuffix

```
5297 \newcommand*{\glxtrscsuffix}{\glstextup{\glspluralsuffix}}
```

long-short-sc

```
5298 \newabbreviationstyle{long-short-sc}%  
5299 {%  
5300   \GlsXtrUseAbbrStyleSetup{long-short}%  
5301 }%  
5302 {%
```

Mostly as long-short style:

```
5303   \GlsXtrUseAbbrStyleFmts{long-short}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5304   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5305   \renewcommand*{\glabbrvfont[1]{\glxtrscfont{##1}}}%  
5306   \renewcommand*{\glfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%  
5307 }
```

g-short-sc-desc

```
5308 \newabbreviationstyle{long-short-sc-desc}%  
5309 {%  
5310   \GlsXtrUseAbbrStyleSetup{long-short-desc}%  
5311 }%  
5312 {%
```

Mostly as long-short-desc style:

```
5313   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5314   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5315   \renewcommand*{\glabbrvfont[1]{\glxtrscfont{##1}}}%  
5316   \renewcommand*{\glfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%  
5317 }
```

Now the short (long) version

```
5318 \newabbreviationstyle{short-sc-long}%  
5319 {%  
5320   \GlsXtrUseAbbrStyleSetup{short-long}%  
5321 }%  
5322 {%
```

Mostly as short-long style:

```
5323   \GlsXtrUseAbbrStyleFmts{short-long}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5324 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5325 \renewcommand*{\glssabbrvfont[1]{\glxtrscfont{##1}}}%
5326 \renewcommand*{\glssfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%
5327 }
```

As before but user provides description

```
5328 \newabbreviationstyle{short-sc-long-desc}%
5329 {%
5330 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5331 }%
5332 {%
```

Mostly as short-long-desc style:

```
5333 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5334 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5335 \renewcommand*{\glssabbrvfont[1]{\glxtrscfont{##1}}}%
5336 \renewcommand*{\glssfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%
5337 }
```

short-sc

```
5338 \newabbreviationstyle{short-sc}%
5339 {%
5340 \GlsXtrUseAbbrStyleSetup{short-nolong}%
5341 }%
5342 {%
```

Mostly as short style:

```
5343 \GlsXtrUseAbbrStyleFmts{short-nolong}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5344 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5345 \renewcommand*{\glssabbrvfont[1]{\glxtrscfont{##1}}}%
5346 \renewcommand*{\glssfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%
5347 }
```

short-sc-nolong

```
5348 \letabbreviationstyle{short-sc-nolong}{short-sc}
```

short-sc-desc

```
5349 \newabbreviationstyle{short-sc-desc}%
5350 {%
5351 \GlsXtrUseAbbrStyleSetup{short-desc}%
5352 }%
5353 {%
```

Mostly as short style:

```
5354 \GlsXtrUseAbbrStyleFmts{short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5355 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5356 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5357 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
5358 }
```

-sc-nolong-desc

```
5359 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}
```

long-noshort-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like \glssshort.

```
5360 \newabbreviationstyle{long-noshort-sc}%
5361 {%
5362 \GlsXtrUseAbbrStyleSetup{long-noshort}%
5363 }%
5364 {%
```

Mostly as long style:

```
5365 \GlsXtrUseAbbrStyleFmts{long-noshort}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5366 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5367 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5368 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
5369 }
```

long-sc Backward compatibility:

```
5370 \@glxtr@deprecated@abbrstyle{long-sc}{long-noshort-sc}
```

noshort-sc-desc The smallcaps font will only be used if the short form is explicitly invoked through commands like \glssshort.

```
5371 \newabbreviationstyle{long-noshort-sc-desc}%
5372 {%
5373 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5374 }%
5375 {%
```

Mostly as long style:

```
5376 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5377 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5378 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5379 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
5380 }
```

long-desc-sc Backward compatibility:

```
5381 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}
```

ort-sc-footnote

```
5382 \newabbreviationstyle{short-sc-footnote}%  
5383 {%  
5384   \GlsXtrUseAbbrStyleSetup{short-footnote}%  
5385 }%  
5386 {%
```

Mostly as long style:

```
5387   \GlsXtrUseAbbrStyleFmts{short-footnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5388   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5389   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%  
5390   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%  
5391 }
```

footnote-sc Backward compatibility:

```
5392 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}
```

sc-postfootnote

```
5393 \newabbreviationstyle{short-sc-postfootnote}%  
5394 {%  
5395   \GlsXtrUseAbbrStyleSetup{short-postfootnote}%  
5396 }%  
5397 {%
```

Mostly as long style:

```
5398   \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5399   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5400   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%  
5401   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%  
5402 }
```

postfootnote-sc Backward compatibility:

```
5403 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}
```

1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the relsize package, which must be loaded by the user. These styles all use:

\glxtrsmfont

```
5404 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}
```

sxtrfirstsmfont

```
5405 \newcommand*{\glxtrfirstsmfont}[1]{\glxtrsmfont{#1}}
```

and for the default short form suffix:

```

\glxtrmsuffix
5406 \newcommand*{\glxtrmsuffix}{\glspluralsuffix}

long-short-sm
5407 \newabbreviationstyle{long-short-sm}%
5408 {%
5409   \GlsXtrUseAbbrStyleSetup{long-short}%
5410 }%
5411 {%
    Mostly as long-short style:
5412   \GlsXtrUseAbbrStyleFmts{long-short}%
5413   \renewcommand*{\glssabrvfont[1]{\glxtrsmfont{##1}}}%
5414   \renewcommand*{\glsfirstabrvfont[1]{\glxtrfirstsmfont{##1}}}%
5415   \renewcommand*{\abrvpluralsuffix}{\protect\glxtrmsuffix}%
5416 }

g-short-sm-desc
5417 \newabbreviationstyle{long-short-sm-desc}%
5418 {%
5419   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5420 }%
5421 {%
    Mostly as long-short-desc style:
5422   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
5423   \renewcommand*{\glssabrvfont[1]{\glxtrsmfont{##1}}}%
5424   \renewcommand*{\glsfirstabrvfont[1]{\glxtrfirstsmfont{##1}}}%
5425   \renewcommand*{\abrvpluralsuffix}{\protect\glxtrmsuffix}%
5426 }

short-sm-long   Now the short (long) version
5427 \newabbreviationstyle{short-sm-long}%
5428 {%
5429   \GlsXtrUseAbbrStyleSetup{short-long}%
5430 }%
5431 {%
    Mostly as short-long style:
5432   \GlsXtrUseAbbrStyleFmts{short-long}%
5433   \renewcommand*{\glssabrvfont[1]{\glxtrsmfont{##1}}}%
5434   \renewcommand*{\glsfirstabrvfont[1]{\glxtrfirstsmfont{##1}}}%
5435   \renewcommand*{\abrvpluralsuffix}{\protect\glxtrmsuffix}%
5436 }

rt-sm-long-desc  As before but user provides description
5437 \newabbreviationstyle{short-sm-long-desc}%
5438 {%
5439   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5440 }%
5441 {%

```

Mostly as short-long-desc style:

```
5442 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
5443 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
5444 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
5445 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
5446 }
```

short-sm

```
5447 \newabbreviationstyle{short-sm}%
5448 {%
5449 \GlsXtrUseAbbrStyleSetup{short-nolong}%
5450 }%
5451 {%
```

Mostly as short style:

```
5452 \GlsXtrUseAbbrStyleFmts{short-nolong}%
5453 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
5454 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
5455 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
5456 }
```

short-sm-nolong

```
5457 \letabbreviationstyle{short-sm-nolong}{short-sm}
```

short-sm-desc

```
5458 \newabbreviationstyle{short-sm-desc}%
5459 {%
5460 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
5461 }%
5462 {%
```

Mostly as short style:

```
5463 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
5464 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
5465 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
5466 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
5467 }
```

-sm-nolong-desc

```
5468 \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`.

```
5469 \newabbreviationstyle{long-noshort-sm}%
5470 {%
5471 \GlsXtrUseAbbrStyleSetup{long-noshort}%
5472 }%
5473 {%
```

Mostly as long style:

```
5474 \GlsXtrUseAbbrStyleFmts{long-noshort}%  
5475 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%  
5476 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%  
5477 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%  
5478 }
```

long-sm Backward compatibility:

```
5479 \@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.

```
5480 \newabbreviationstyle{long-noshort-sm-desc}%  
5481 {%  
5482 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%  
5483 }%  
5484 {%
```

Mostly as long style:

```
5485 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%  
5486 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%  
5487 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%  
5488 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%  
5489 }
```

long-desc-sm Backward compatibility:

```
5490 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}
```

short-sm-footnote

```
5491 \newabbreviationstyle{short-sm-footnote}%  
5492 {%  
5493 \GlsXtrUseAbbrStyleSetup{short-footnote}%  
5494 }%  
5495 {%
```

Mostly as long style:

```
5496 \GlsXtrUseAbbrStyleFmts{short-footnote}%  
5497 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%  
5498 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%  
5499 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%  
5500 }
```

footnote-sm Backward compatibility:

```
5501 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}
```

sm-postfootnote

```
5502 \newabbreviationstyle{short-sm-postfootnote}%  
5503 {%  
5504 \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
```

5505 }%
 5506 {%

Mostly as long style:

5507 \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
 5508 \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
 5509 \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}%
 5510 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
 5511 }

postfootnote-sm Backward compatibility:

5512 \@glstr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

1.6.5 Predefined Styles (Emphasized)

These styles use \emph for the short form.

\glsabbrvemfont

5513 \newcommand*\glsabbrvemfont[1]{\emph{#1}}%

irstabbrvemfont

5514 \newcommand*\glsfirstabbrvemfont[1]{\glsabbrvemfont{#1}}%

firstlongemfont Only used by the “long-em” styles.

5515 \newcommand*\glsfirstlongemfont[1]{\glslongemfont{#1}}%

\glslongemfont Only used by the “long-em” styles.

5516 \newcommand*\glslongemfont[1]{\emph{#1}}%

long-short-em

5517 \newabbreviationstyle{long-short-em}%
 5518 {%
 5519 \GlsXtrUseAbbrStyleSetup{long-short}%
 5520 }%
 5521 {%

Mostly as long-short style:

5522 \GlsXtrUseAbbrStyleFmts{long-short}%
 5523 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
 5524 }

g-short-em-desc

5525 \newabbreviationstyle{long-short-em-desc}%
 5526 {%
 5527 \GlsXtrUseAbbrStyleSetup{long-short-desc}%
 5528 }%
 5529 {%

Mostly as long-short-desc style:

```
5530 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
5531 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5532 }
```

ong-em-short-em

```
5533 \newabbreviationstyle{long-em-short-em}%
5534 {%
```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```
5535 \renewcommand*{\CustomAbbreviationFields}{%
5536   name={\protect\glsabbrvfont{\the\glsshorttok}},
5537   sort={\the\glsshorttok},
5538   first={\protect\glsfirstlongfont{\the\glslongtok}%
5539     \protect\glsxtrfullsep{\the\glslabeltok}%
5540     (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
5541   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
5542     \protect\glsxtrfullsep{\the\glslabeltok}%
5543     (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%
5544   plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
5545   description={\protect\glslongemfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
5546 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5547   \glshasattribute{\the\glslabeltok}{regular}%
5548   {%
5549     \glssetattribute{\the\glslabeltok}{regular}{false}%
5550   }%
5551   {}%
5552 }%
5553 }%
5554 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5555 \GlsXtrUseAbbrStyleFmts{long-short}%
5556 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5557 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5558 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5559 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5560 }
```

m-short-em-desc

```
5561 \newabbreviationstyle{long-em-short-em-desc}%
5562 {%
5563   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5564 }%
5565 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5566 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
```

```

5567 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5568 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5569 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5570 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5571 }

```

short-em-long Now the short (long) version

```

5572 \newabbreviationstyle{short-em-long}%
5573 {%
5574   \GlsXtrUseAbbrStyleSetup{short-long}%
5575 }%
5576 {%

```

Mostly as short-long style:

```

5577   \GlsXtrUseAbbrStyleFmts{short-long}%
5578   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5579   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5580 }

```

short-em-long-desc As before but user provides description

```

5581 \newabbreviationstyle{short-em-long-desc}%
5582 {%
5583   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5584 }%
5585 {%

```

Mostly as short-long-desc style:

```

5586   \GlsXtrUseAbbrStyleFmts{short-long-desc}%
5587   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5588   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5589   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5590   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5591 }

```

short-em-long-em

```

5592 \newabbreviationstyle{short-em-long-em}%
5593 {%

```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```

5594   \renewcommand*{\CustomAbbreviationFields}{%
5595     name={\protect\glsabbrvfont{\the\glsshorttok}},
5596     sort={\the\glsshorttok},
5597     description={\protect\glslongemfont{\the\glslongtok}},%
5598     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
5599     \protect\glsxtrfullsep{\the\glslabeltok}%
5600     (\protect\glsfirstlongfont{\the\glslongtok}}),%
5601     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
5602     \protect\glsxtrfullsep{\the\glslabeltok}%
5603     (\protect\glsfirstlongfont{\the\glslongpltok}}),%
5604     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```
5605 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5606   \glshasattribute{\the\glslabeltok}{regular}%
5607   {%
5608     \glsselattribute{\the\glslabeltok}{regular}{false}%
5609   }%
5610   {}%
5611 }%
5612 }%
5613 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5614 \GlsXtrUseAbbrStyleFmts{short-long}%
5615 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
5616 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
5617 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5618 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5619 }
```

em-long-em-desc

```
5620 \newabbreviationstyle{short-em-long-em-desc}%
5621 {%
5622   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5623 }%
5624 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5625 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
5626 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
5627 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
5628 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5629 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5630 }
```

short-em

```
5631 \newabbreviationstyle{short-em}%
5632 {%
5633   \GlsXtrUseAbbrStyleSetup{short-nolong}%
5634 }%
5635 {%
```

Mostly as short style:

```
5636 \GlsXtrUseAbbrStyleFmts{short-nolong}%
5637 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
5638 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
5639 }
```

short-em-nolong

```
5640 \letabbreviationstyle{short-em-nolong}{short-em}
```

```

short-em-desc
5641 \newabbreviationstyle{short-em-desc}%
5642 {%
5643   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
5644 }%
5645 {%

  Mostly as short style:
5646   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
5647   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5648   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5649 }

-em-nolong-desc
5650 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

long-noshort-em  The short form is explicitly invoked through commands like \glsshort.
5651 \newabbreviationstyle{long-noshort-em}%
5652 {%
5653   \GlsXtrUseAbbrStyleSetup{long-noshort}%
5654 }%
5655 {%

  Mostly as long-noshort style:
5656   \GlsXtrUseAbbrStyleFmts{long-noshort}%
5657   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5658   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5659 }

long-em  Backward compatibility:
5660 \@glsxtr@deprecated@abbrstyle{long-em}{long-noshort-em}

g-em-noshort-em  The short form is explicitly invoked through commands like \glsshort.
5661 \newabbreviationstyle{long-em-noshort-em}%
5662 {%
5663   \renewcommand*\CustomAbbreviationFields{%
5664     name={\protect\glsabbrvfont{\the\glsshorttok}},
5665     sort={\the\glsshorttok},
5666     first={\protect\glsfirstlongfont{\the\glslongtok}},
5667     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5668     text={\the\glslongtok},
5669     plural={\the\glslongpltok},%
5670     description={\protect\glslongemfont{\the\glslongtok}}%
5671   }%
5672   \renewcommand*\GlsXtrPostNewAbbreviation{%
5673     \glssetAttribute{\the\glslabeltok}{regular}{true}}%
5674 }%
5675 {%

```

Mostly as long-noshort style:

```
5676 \GlsXtrUseAbbrStyleFmts{long-noshort}%
5677 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5678 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5679 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
5680 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
5681 }
```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like \glsshort.

```
5682 \newabbreviationstyle{long-noshort-em-desc}%
5683 {%
5684 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5685 }%
5686 {%
```

Mostly as long style:

```
5687 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
5688 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5689 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5690 }
```

long-desc-em Backward compatibility:

```
5691 \@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.

```
5692 \newabbreviationstyle{long-em-noshort-em-desc}%
5693 {%
5694 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5695 }%
5696 {%
```

Mostly as long style:

```
5697 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
5698 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5699 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5700 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
5701 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
5702 }
```

ort-em-footnote

```
5703 \newabbreviationstyle{short-em-footnote}%
5704 {%
5705 \GlsXtrUseAbbrStyleSetup{short-footnote}%
5706 }%
5707 {%
```

Mostly as long style:

```
5708 \GlsXtrUseAbbrStyleFmts{short-footnote}%
5709 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5710 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5711 }
```

footnote-em Backward compatibility:

```
5712 \@glsxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}
```

em-postfootnote

```
5713 \newabbreviationstyle{short-em-postfootnote}%
5714 {%
5715 \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
5716 }%
5717 {%
```

Mostly as long style:

```
5718 \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
5719 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5720 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5721 }
```

postfootnote-em Backward compatibility:

```
5722 \@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.

```
5723 \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.

```
5724 \ifdef\glscurrentfieldvalue
5725 {
5726 \newcommand*\{glsxtruserparen}[2]{%
5727 \glsxtrfullsep{#2}%
5728 (#1\ifglshasfield{glsxtruserfield}{#2}{, \glscurrentfieldvalue}{})%
5729 }
5730 }
5731 {
5732 \newcommand*\{glsxtruserparen}[2]{%
5733 \glsxtrfullsep{#2}%
5734 (#1\ifglshasfield{glsxtruserfield}{#2}{, \@glo@thisvalue}{})%
5735 }
5736 }
```

Font used for short form:

lsabbrvuserfont

```
5737 \newcommand*{\glsabbrvuserfont}[1]{#1}
```

Font used for short form on first use:

stabbrvuserfont

```
5738 \newcommand*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}
```

Font used for long form:

glslonguserfont

```
5739 \newcommand*{\glslonguserfont}[1]{#1}
```

Font used for long form on first use:

rstlonguserfont

```
5740 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
5741 \newcommand*{\glsxtrusersuffix}{\glspluralsuffix}
```

long-short-user

```
5742 \newabbreviationstyle{long-short-user}%  
5743 {%
```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```
5744 \renewcommand*{\CustomAbbreviationFields}{%  
5745   name={\protect\glsabbrvfont{\the\glsshorttok}},  
5746   sort={\the\glsshorttok},  
5747   first={\protect\glsfirstlongfont{\the\glslongtok}}%  
5748   \protect\glsxtruserparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}{\the\glslabeltok}},  
5749   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%  
5750   \protect\glsxtruserparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}{\the\glslabeltok}}%  
5751   plural={\protect\glsabrvfont{\the\glsshortpltok}},%  
5752   description={\protect\glslonguserfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
5753 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
5754   \glshasattribute{\the\glslabeltok}{regular}%  
5755   {%  
5756     \glssetattribute{\the\glslabeltok}{regular}{false}%  
5757   }%  
5758   {}%  
5759 }%  
5760 }%  
5761 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```

5762 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
5763 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
5764 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
5765 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
5766 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5767 \renewcommand*{\glxtrfullformat}[2]{%
5768   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
5769   \ifglxtrininsertinside\else##2\fi
5770   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
5771 }%
5772 \renewcommand*{\glxtrfullplformat}[2]{%
5773   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
5774   \ifglxtrininsertinside\else##2\fi
5775   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
5776 }%
5777 \renewcommand*{\Glsxtrfullformat}[2]{%
5778   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
5779   \ifglxtrininsertinside\else##2\fi
5780   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
5781 }%
5782 \renewcommand*{\Glsxtrfullplformat}[2]{%
5783   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
5784   \ifglxtrininsertinside\else##2\fi
5785   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
5786 }%
5787 }

```

short-user-desc

```

5788 \newabbreviationstyle{long-short-user-desc}%
5789 {%
5790   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5791 }%
5792 {%
5793   \GlsXtrUseAbbrStyleFmts{long-short-user}%
5794 }

```

short-long-user

```

5795 \newabbreviationstyle{short-long-user}%
5796 {%

```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```

5797 \renewcommand*{\CustomAbbreviationFields}{%
5798   name={\protect\glsabbrvfont{\the\glsshorttok}},
5799   sort={\the\glsshorttok},
5800   description={\protect\glslonguserfont{\the\glslongtok}},%
5801   first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
5802   \protect\glxtruserparen{\protect\glslonguserfont{\the\glslongtok}}{\the\glslabeltok}},%

```

```

5803 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
5804 \protect\glsxtruserparen{\protect\glslonguserfont{\the\glslongpltok}}{\the\glslabeltok}},%
5805 plural={\protect\glsabbvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

5806 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5807 \glshasattribute{\the\glslabeltok}{regular}%
5808 {%
5809 \glssetattribute{\the\glslabeltok}{regular}{false}%
5810 }%
5811 {}%
5812 }%
5813 }%
5814 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5815 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
5816 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvuserfont{##1}}%
5817 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
5818 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
5819 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5820 \renewcommand*{\glsxtrfullformat}[2]{%
5821 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5822 \ifglxtrinsertinside\else##2\fi
5823 \glsxtruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%
5824 }%
5825 \renewcommand*{\glsxtrfullplformat}[2]{%
5826 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5827 \ifglxtrinsertinside\else##2\fi
5828 \glsxtruserparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}{##1}%
5829 }%
5830 \renewcommand*{\Glsxtrfullformat}[2]{%
5831 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5832 \ifglxtrinsertinside\else##2\fi
5833 \glsxtruserparen{\glsfirstlongfont{\Glsaccesslong{##1}}}{##1}%
5834 }%
5835 \renewcommand*{\Glsxtrfullplformat}[2]{%
5836 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5837 \ifglxtrinsertinside\else##2\fi
5838 \glsxtruserparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}{##1}%
5839 }%
5840 }

```

-long-user-desc

```

5841 \newabbreviationstyle{short-long-user-desc}%
5842 {%
5843 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5844 }%

```

```

5845 {%
5846   \GlsXtrUseAbbrStyleFmts{short-long-user}%
5847 }

```

1.7 Using Entries in Headings

There are four main problems with using entries in sectioning commands: they can mess with the first use flag if they end up in the table of contents, they can add unwanted numbers to the entry's location list, the label is corrupted if used inside `\MakeUppercase` (which is used by the default headings style) and they need to be expandable for PDF bookmarks. The glossaries package therefore recommends the use of the expandable commands, such as `\glstryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glssabrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref`'s `\texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glstrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to `false`, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that `glossaries` automatically loads `textcase`, so the label can be protected from case change with `textcase`'s `\NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```

5848 \let\@glstr@org@markright\markright

```

Redefine (grouping not added in case it interferes with the original code):

```

5849 \renewcommand*{\markright}[1]{%
5850   \glstrmarkhook
5851   \@glstr@org@markright{\@glstrinmark#1\@glstrnotinmark}%
5852   \glstrrestoremarkhook
5853 }

```

`\markboth` Save original definition:

```

5854 \let\@glstr@org@markboth\markboth

```

Redefine (grouping not added in case it interferes with the original code):

```

5855 \renewcommand*{\markboth}[2]{%
5856   \glstrmarkhook
5857   \@glstr@org@markboth
5858   {\@glstrinmark#1\@glstrnotinmark}%

```

```

5859   {\@glsxtrinmark#2\@glsxtrnotinmark}%
5860 \glsxtrrestoremarkhook
5861 }

```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```

5862 \newcommand*{\glsxtrRevertMarks}{%
5863   \let\markright\@glsxtr@org@markright
5864   \let\markboth\@glsxtr@org@markboth
5865 }

```

\glsxtrifinmark

```

5866 \newcommand*{\glsxtrifinmark}[2]{#2}

```

\@glsxtrinmark

```

5867 \newrobustcmd*{\@glsxtrinmark}{%
5868   \let\glsxtrifinmark\@firstoftwo
5869 }

```

glsxtrnotinmark

```

5870 \newrobustcmd*{\@glsxtrnotinmark}{%
5871   \let\glsxtrifinmark\@secondoftwo
5872 }

```

\glsxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```

5873 \newcommand*{\glsxtrmarkhook}{%

```

Save current definitions:

```

5874   \let\@glsxtr@org@MakeUppercase\MakeUppercase
5875   \let\@glsxtr@org@glsxtrtitleshort\glsxtrtitleshort
5876   \let\@glsxtr@org@glsxtrtitleshortpl\glsxtrtitleshortpl
5877   \let\@glsxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
5878   \let\@glsxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
5879   \let\@glsxtr@org@glsxtrtitletext\glsxtrtitletext
5880   \let\@glsxtr@org@Glsxtrtitletext\Glsxtrtitletext
5881   \let\@glsxtr@org@glsxtrtitleplural\glsxtrtitleplural
5882   \let\@glsxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
5883   \let\@glsxtr@org@glsxtrtitlefirst\glsxtrtitlefirst
5884   \let\@glsxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
5885   \let\@glsxtr@org@glsxtrtitlefirstplural\glsxtrtitlefirstplural
5886   \let\@glsxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
5887   \let\@glsxtr@org@glsxtrtitlelong\glsxtrtitlelong
5888   \let\@glsxtr@org@glsxtrtitlelongpl\glsxtrtitlelongpl
5889   \let\@glsxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
5890   \let\@glsxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
5891   \let\@glsxtr@org@glsxtrtitlefull\glsxtrtitlefull
5892   \let\@glsxtr@org@glsxtrtitlefullpl\glsxtrtitlefullpl

```

```

5893 \let\@glsxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
5894 \let\@glsxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl

```

New definitions

```

5895 \let\glsxtrifinmark\@firstoftwo
5896 \let\MakeUppercase\MakeTextUppercase
5897 \let\glsxtrtitleshort\glsxtrheadshort
5898 \let\glsxtrtitleshortpl\glsxtrheadshortpl
5899 \let\Glsxtrtitleshort\Glsxtrheadshort
5900 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
5901 \let\glsxtrtitletext\glsxtrheadtext
5902 \let\Glsxtrtitletext\Glsxtrheadtext
5903 \let\glsxtrtitleplural\glsxtrheadplural
5904 \let\Glsxtrtitleplural\Glsxtrheadplural
5905 \let\glsxtrtitlefirst\glsxtrheadfirst
5906 \let\Glsxtrtitlefirst\Glsxtrheadfirst
5907 \let\glsxtrtitlefirstplural\glsxtrheadfirstplural
5908 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
5909 \let\glsxtrtitlelong\glsxtrheadlong
5910 \let\glsxtrtitlelongpl\glsxtrheadlongpl
5911 \let\Glsxtrtitlelong\Glsxtrheadlong
5912 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
5913 \let\glsxtrtitlefull\glsxtrheadfull
5914 \let\glsxtrtitlefullpl\glsxtrheadfullpl
5915 \let\Glsxtrtitlefull\Glsxtrheadfull
5916 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
5917 }

```

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.)

```

5918 \newcommand*{\glsxtrrestoremarkhook}{%
5919 \let\glsxtrifinmark\@secondoftwo
5920 \let\MakeUppercase\@glsxtr@org@MakeUppercase
5921 \let\glsxtrtitleshort\@glsxtr@org@glsxtrtitleshort
5922 \let\glsxtrtitleshortpl\@glsxtr@org@glsxtrtitleshortpl
5923 \let\Glsxtrtitleshort\@glsxtr@org@Glsxtrtitleshort
5924 \let\Glsxtrtitleshortpl\@glsxtr@org@Glsxtrtitleshortpl
5925 \let\glsxtrtitletext\@glsxtr@org@glsxtrtitletext
5926 \let\Glsxtrtitletext\@glsxtr@org@Glsxtrtitletext
5927 \let\glsxtrtitleplural\@glsxtr@org@glsxtrtitleplural
5928 \let\Glsxtrtitleplural\@glsxtr@org@Glsxtrtitleplural
5929 \let\glsxtrtitlefirst\@glsxtr@org@glsxtrtitlefirst
5930 \let\Glsxtrtitlefirst\@glsxtr@org@Glsxtrtitlefirst
5931 \let\glsxtrtitlefirstplural\@glsxtr@org@glsxtrtitlefirstplural
5932 \let\Glsxtrtitlefirstplural\@glsxtr@org@Glsxtrtitlefirstplural
5933 \let\glsxtrtitlelong\@glsxtr@org@glsxtrtitlelong
5934 \let\glsxtrtitlelongpl\@glsxtr@org@glsxtrtitlelongpl

```

```

5935 \let\Glsxtrtitlelong\@glsxtr@org@Glsxtrtitlelong
5936 \let\Glsxtrtitlelongpl\@glsxtr@org@Glsxtrtitlelongpl
5937 \let\glsxtrtitlefull\@glsxtr@org@glsxtrtitlefull
5938 \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
5939 \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
5940 \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl
5941 }

```

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.

```

5942 \newcommand*{\glsxtrheadshort}[1]{%
5943   \protect\NoCaseChange
5944   {%
5945     \glsifattribute{#1}{headuc}{true}%
5946     {%
5947       \GLSxtrshort[noindex,hyper=false]{#1}[]%
5948     }%
5949     {%
5950       \glsxtrshort[noindex,hyper=false]{#1}[]%
5951     }%
5952   }%
5953 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

5954 \newrobustcmd*{\lsxtrtitleshort}[1]{%
5955   \glsxtrshort[noindex,hyper=false]{#1}[]%
5956 }

```

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.

```

5957 \newcommand*{\glsxtrheadshortpl}[1]{%
5958   \protect\NoCaseChange
5959   {%
5960     \glsifattribute{#1}{headuc}{true}%
5961     {%
5962       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
5963     }%
5964     {%
5965       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
5966     }%
5967   }%
5968 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```

5969 \newrobustcmd*{\lsxtrtitleshortpl}[1]{%

```

```

5970 \glsxtrshortpl[noindex,hyper=false]{#1}[]%
5971 }

```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```

5972 \newcommand*{\Glsxtrheadshort}[1]{%
5973 \protect\NoCaseChange
5974 {%
5975 \glsifattribute{#1}{headuc}{true}%
5976 {%
5977 \Glsxtrshort[noindex,hyper=false]{#1}[]%
5978 }%
5979 {%
5980 \Glsxtrshort[noindex,hyper=false]{#1}[]%
5981 }%
5982 }%
5983 }

```

ltxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5984 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
5985 \Glsxtrshort[noindex,hyper=false]{#1}[]%
5986 }

```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```

5987 \newcommand*{\Glsxtrheadshortpl}[1]{%
5988 \protect\NoCaseChange
5989 {%
5990 \glsifattribute{#1}{headuc}{true}%
5991 {%
5992 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
5993 }%
5994 {%
5995 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
5996 }%
5997 }%
5998 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5999 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
6000 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
6001 }

```

\glsxtrheadtext As above but for the text value.

```

6002 \newcommand*{\glsxtrheadtext}[1]{%
6003 \protect\NoCaseChange

```

```

6004 {%
6005   \glsifattribute{#1}{headuc}{true}%
6006   {%
6007     \GLStext[noindex,hyper=false]{#1}[]%
6008   }%
6009   {%
6010     \glstext[noindex,hyper=false]{#1}[]%
6011   }%
6012 }%
6013 }

```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```

6014 \newrobustcmd*{\glsxtrtitletext}[1]{%
6015   \glstext[noindex,hyper=false]{#1}[]%
6016 }

```

`\Glsxtrheadtext` First letter converted to upper case

```

6017 \newcommand*{\Glsxtrheadtext}[1]{%
6018   \protect\NoCaseChange
6019   {%
6020     \glsifattribute{#1}{headuc}{true}%
6021     {%
6022       \GLStext[noindex,hyper=false]{#1}[]%
6023     }%
6024     {%
6025       \Glstext[noindex,hyper=false]{#1}[]%
6026     }%
6027   }%
6028 }

```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```

6029 \newrobustcmd*{\Glsxtrtitletext}[1]{%
6030   \Glstext[noindex,hyper=false]{#1}[]%
6031 }

```

`lsxtrheadplural` As above but for the plural value.

```

6032 \newcommand*{\lsxtrheadplural}[1]{%
6033   \protect\NoCaseChange
6034   {%
6035     \glsifattribute{#1}{headuc}{true}%
6036     {%
6037       \GLSplural[noindex,hyper=false]{#1}[]%
6038     }%
6039     {%
6040       \glsplural[noindex,hyper=false]{#1}[]%
6041     }%
6042   }%
6043 }

```

sxtrtitleplural Command to display plural value in section title and table of contents.

```
6044 \newrobustcmd*{\glsxtrtitleplural}[1]{%
6045   \glsplural[noindex,hyper=false]{#1}[]%
6046 }
```

lsxtrheadplural Convert first letter to upper case.

```
6047 \newcommand*{\Glsxtrheadplural}[1]{%
6048   \protect\NoCaseChange
6049   {%
6050     \glusifattribute{#1}{headuc}{true}%
6051     {%
6052       \GLSplural[noindex,hyper=false]{#1}[]%
6053     }%
6054     {%
6055       \Glsplural[noindex,hyper=false]{#1}[]%
6056     }%
6057   }%
6058 }
```

sxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
6059 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
6060   \Glsplural[noindex,hyper=false]{#1}[]%
6061 }
```

glsxtrheadfirst As above but for the first value.

```
6062 \newcommand*{\glsxtrheadfirst}[1]{%
6063   \protect\NoCaseChange
6064   {%
6065     \glusifattribute{#1}{headuc}{true}%
6066     {%
6067       \GLSfirst[noindex,hyper=false]{#1}[]%
6068     }%
6069     {%
6070       \glsfirst[noindex,hyper=false]{#1}[]%
6071     }%
6072   }%
6073 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```
6074 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
6075   \glsfirst[noindex,hyper=false]{#1}[]%
6076 }
```

Glsxtrheadfirst First letter converted to upper case

```
6077 \newcommand*{\Glsxtrheadfirst}[1]{%
6078   \protect\NoCaseChange
6079   {%
```

```

6080 \glsifattribute{#1}{headuc}{true}%
6081 {%
6082 \GLSfirst[noindex,hyper=false]{#1}[]%
6083 }%
6084 {%
6085 \GLSfirst[noindex,hyper=false]{#1}[]%
6086 }%
6087 }%
6088 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```

6089 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
6090 \GLSfirst[noindex,hyper=false]{#1}[]%
6091 }

```

headfirstplural As above but for the firstplural value.

```

6092 \newcommand*{\Glsxtrheadfirstplural}[1]{%
6093 \protect\NoCaseChange
6094 {%
6095 \glsifattribute{#1}{headuc}{true}%
6096 {%
6097 \GLSfirstplural[noindex,hyper=false]{#1}[]%
6098 }%
6099 {%
6100 \Glsfirstplural[noindex,hyper=false]{#1}[]%
6101 }%
6102 }%
6103 }

```

itlefirstplural Command to display firstplural value in section title and table of contents.

```

6104 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
6105 \Glsfirstplural[noindex,hyper=false]{#1}[]%
6106 }

```

headfirstplural First letter converted to upper case

```

6107 \newcommand*{\Glsxtrheadfirstplural}[1]{%
6108 \protect\NoCaseChange
6109 {%
6110 \glsifattribute{#1}{headuc}{true}%
6111 {%
6112 \GLSfirstplural[noindex,hyper=false]{#1}[]%
6113 }%
6114 {%
6115 \Glsfirstplural[noindex,hyper=false]{#1}[]%
6116 }%
6117 }%
6118 }

```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
6119 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
6120   \Glsfirstplural[noindex,hyper=false]{#1}[]%
6121 }
```

`\glxtrheadlong` Command used to display long form in the page header.

```
6122 \newcommand*{\glxtrheadlong}[1]{%
6123   \protect\NoCaseChange
6124   {%
6125     \glsifattribute{#1}{headuc}{true}%
6126     {%
6127       \GLSxtrlong[noindex,hyper=false]{#1}[]%
6128     }%
6129     {%
6130       \glxtrlong[noindex,hyper=false]{#1}[]%
6131     }%
6132   }%
6133 }
```

`\glxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
6134 \newrobustcmd*{\glxtrtitlelong}[1]{%
6135   \glxtrlong[noindex,hyper=false]{#1}[]%
6136 }
```

`\glxtrheadlongpl` 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.

```
6137 \newcommand*{\glxtrheadlongpl}[1]{%
6138   \protect\NoCaseChange
6139   {%
6140     \glsifattribute{#1}{headuc}{true}%
6141     {%
6142       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
6143     }%
6144     {%
6145       \glxtrlongpl[noindex,hyper=false]{#1}[]%
6146     }%
6147   }%
6148 }
```

`\glxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```
6149 \newrobustcmd*{\glxtrtitlelongpl}[1]{%
6150   \glxtrlongpl[noindex,hyper=false]{#1}[]%
6151 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

6152 \newcommand*{\Glsxtrheadlong}[1]{%
6153   \protect\NoCaseChange
6154   {%
6155     \glsifattribute{#1}{headuc}{true}%
6156     {%
6157       \GLSxtrlong[noindex,hyper=false]{#1}[]%
6158     }%
6159     {%
6160       \Glsxtrlong[noindex,hyper=false]{#1}[]%
6161     }%
6162   }%
6163 }

```

Glsxtrtitlelong Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

6164 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
6165   \Glsxtrlong[noindex,hyper=false]{#1}[]%
6166 }

```

lgsxtrheadlongpl Command used to display plural long form in the page header with the first letter converted to upper case.

```

6167 \newcommand*{\Glsxtrheadlongpl}[1]{%
6168   \protect\NoCaseChange
6169   {%
6170     \glsifattribute{#1}{headuc}{true}%
6171     {%
6172       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
6173     }%
6174     {%
6175       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
6176     }%
6177   }%
6178 }

```

gsxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

6179 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
6180   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
6181 }

```

\glsxtrheadfull Command used to display full form in the page header.

```

6182 \newcommand*{\glsxtrheadfull}[1]{%
6183   \protect\NoCaseChange
6184   {%
6185     \glsifattribute{#1}{headuc}{true}%
6186     {%
6187       \GLSxtrfull[noindex,hyper=false]{#1}[]%
6188     }%

```

```

6189   {%
6190     \glsxtrfull[noindex,hyper=false]{#1}[]%
6191   }%
6192 }%
6193 }

```

`\glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents.

```

6194 \newrobustcmd*{\glsxtrtitlefull}[1]{%
6195   \glsxtrfull[noindex,hyper=false]{#1}[]%
6196 }

```

`\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.

```

6197 \newcommand*{\glsxtrheadfullpl}[1]{%
6198   \protect\NoCaseChange
6199   {%
6200     \glsifattribute{#1}{headuc}{true}%
6201     {%
6202       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
6203     }%
6204     {%
6205       \glsxtrfullpl[noindex,hyper=false]{#1}[]%
6206     }%
6207   }%
6208 }

```

`\glsxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents.

```

6209 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
6210   \glsxtrfullpl[noindex,hyper=false]{#1}[]%
6211 }

```

`\GLSxtrheadfull` Command used to display full form in the page header with the first letter converted to upper case.

```

6212 \newcommand*{\GLSxtrheadfull}[1]{%
6213   \protect\NoCaseChange
6214   {%
6215     \glsifattribute{#1}{headuc}{true}%
6216     {%
6217       \GLSxtrfull[noindex,hyper=false]{#1}[]%
6218     }%
6219     {%
6220       \GLSxtrfull[noindex,hyper=false]{#1}[]%
6221     }%
6222   }%
6223 }

```

`\GLSxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

6224 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
6225   \Glsxtrfull[noindex,hyper=false]{#1}[]%
6226 }

```

`\lsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```

6227 \newcommand*{\Glsxtrheadfullpl}[1]{%
6228   \protect\NoCaseChange
6229   {%
6230     \glsifattribute{#1}{headuc}{true}%
6231     {%
6232       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
6233     }%
6234     {%
6235       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
6236     }%
6237   }%
6238 }

```

`\sxttrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

6239 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
6240   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
6241 }

```

`\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.

```

6242 \ifdef\texorpdfstring
6243 {
6244   \newcommand*{\glsfmtshort}[1]{%
6245     \texorpdfstring
6246     {\glsxtrtitleshort{#1}}%
6247     {\glsentryshort{#1}}%
6248   }
6249 }
6250 {
6251   \newcommand*{\glsfmtshort}[1]{%
6252     \glsxtrtitleshort{#1}}
6253 }

```

Similarly for the plural version.

`\glsfmtshortpl`

```

6254 \ifdef\texorpdfstring
6255 {
6256   \newcommand*{\glsfmtshortpl}[1]{%
6257     \texorpdfstring
6258     {\glsxtrtitleshortpl{#1}}%
6259     {\glsentryshortpl{#1}}%

```

```

6260 }
6261 }
6262 {
6263   \newcommand*{\glsfmtshortpl}[1]{%
6264     \glsxtrtitleshortpl{#1}}
6265 }

```

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).

```

6266 \ifdef\texorpdfstring
6267 {
6268   \newcommand*{\Glsfmtshort}[1]{%
6269     \texorpdfstring
6270       {\Glsxtrtitleshort{#1}}%
6271     {\glsentryshort{#1}}%
6272   }
6273 }
6274 {
6275   \newcommand*{\Glsfmtshort}[1]{%
6276     \Glsxtrtitleshort{#1}}
6277 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

6278 \ifdef\texorpdfstring
6279 {
6280   \newcommand*{\Glsfmtshortpl}[1]{%
6281     \texorpdfstring
6282       {\Glsxtrtitleshortpl{#1}}%
6283     {\glsentryshortpl{#1}}%
6284   }
6285 }
6286 {
6287   \newcommand*{\Glsfmtshortpl}[1]{%
6288     \Glsxtrtitleshortpl{#1}}
6289 }

```

`\glsfmttext` As above but for the text value.

```

6290 \ifdef\texorpdfstring
6291 {
6292   \newcommand*{\glsfmttext}[1]{%
6293     \texorpdfstring
6294       {\glsxtrtitletext{#1}}%
6295     {\glsentrytext{#1}}%
6296   }
6297 }
6298 {
6299   \newcommand*{\glsfmttext}[1]{%

```

```

6300 \glstrtitletext{#1}}
6301 }

```

`\Glsfmttext` First letter converted to upper case.

```

6302 \ifdef\teorpdfstring
6303 {
6304 \newcommand*\Glsfmttext}[1]{%
6305 \teorpdfstring
6306 {\Glsxtrtitletext{#1}}%
6307 {\glstrytext{#1}}%
6308 }
6309 }
6310 {
6311 \newcommand*\Glsfmttext}[1]{%
6312 \Glsxtrtitletext{#1}}
6313 }

```

`\glsfmtplural` As above but for the plural value.

```

6314 \ifdef\teorpdfstring
6315 {
6316 \newcommand*\glsfmtplural}[1]{%
6317 \teorpdfstring
6318 {\glstrtitleplural{#1}}%
6319 {\glstryplural{#1}}%
6320 }
6321 }
6322 {
6323 \newcommand*\glsfmtplural}[1]{%
6324 \glstrtitleplural{#1}}
6325 }

```

`\Glsfmtplural` First letter converted to upper case.

```

6326 \ifdef\teorpdfstring
6327 {
6328 \newcommand*\Glsfmtplural}[1]{%
6329 \teorpdfstring
6330 {\Glsxtrtitleplural{#1}}%
6331 {\glstryplural{#1}}%
6332 }
6333 }
6334 {
6335 \newcommand*\Glsfmtplural}[1]{%
6336 \Glsxtrtitleplural{#1}}
6337 }

```

`\glsfmtfirst` As above but for the first value.

```

6338 \ifdef\teorpdfstring
6339 {
6340 \newcommand*\glsfmtfirst}[1]{%

```

```

6341 \texorpdfstring
6342 {\glxtrtitlefirst{#1}}}%
6343 {\glsentryfirst{#1}}}%
6344 }
6345 }
6346 {
6347 \newcommand*{\Glsfmtfirst}[1]{%
6348 \glxtrtitlefirst{#1}}
6349 }

```

\Glsfmtfirst First letter converted to upper case.

```

6350 \ifdef\texorpdfstring
6351 {
6352 \newcommand*{\Glsfmtfirst}[1]{%
6353 \texorpdfstring
6354 {\Glsxtrtitlefirst{#1}}}%
6355 {\glsentryfirst{#1}}}%
6356 }
6357 }
6358 {
6359 \newcommand*{\Glsfmtfirst}[1]{%
6360 \Glsxtrtitlefirst{#1}}
6361 }

```

\glsfmtfirstpl As above but for the firstplural value.

```

6362 \ifdef\texorpdfstring
6363 {
6364 \newcommand*{\glsfmtfirstpl}[1]{%
6365 \texorpdfstring
6366 {\glxtrtitlefirstplural{#1}}}%
6367 {\glsentryfirstplural{#1}}}%
6368 }
6369 }
6370 {
6371 \newcommand*{\glsfmtfirstpl}[1]{%
6372 \glxtrtitlefirstplural{#1}}
6373 }

```

\Glsfmtfirstpl First letter converted to upper case.

```

6374 \ifdef\texorpdfstring
6375 {
6376 \newcommand*{\Glsfmtfirstpl}[1]{%
6377 \texorpdfstring
6378 {\Glsxtrtitlefirstplural{#1}}}%
6379 {\glsentryfirstplural{#1}}}%
6380 }
6381 }
6382 {
6383 \newcommand*{\Glsfmtfirstpl}[1]{%

```

```

6384 \Glsxtrtitlefirstplural{#1}}
6385 }

```

`\glsfmtlong` As above but for the long value.

```

6386 \ifdef\txorpdfstring
6387 {
6388   \newcommand*\glsfmtlong[1]{%
6389     \txorpdfstring
6390     {\glsxtrtitlelong{#1}}%
6391     {\glsentrylong{#1}}%
6392   }
6393 }
6394 {
6395   \newcommand*\glsfmtlong[1]{%
6396     \glsxtrtitlelong{#1}}
6397 }

```

`\Glsfmtlong` First letter converted to upper case.

```

6398 \ifdef\txorpdfstring
6399 {
6400   \newcommand*\Glsfmtlong[1]{%
6401     \txorpdfstring
6402     {\Glsxtrtitlelong{#1}}%
6403     {\glsentrylong{#1}}%
6404   }
6405 }
6406 {
6407   \newcommand*\Glsfmtlong[1]{%
6408     \Glsxtrtitlelong{#1}}
6409 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

6410 \ifdef\txorpdfstring
6411 {
6412   \newcommand*\glsfmtlongpl[1]{%
6413     \txorpdfstring
6414     {\glsxtrtitlelongpl{#1}}%
6415     {\glsentrylongpl{#1}}%
6416   }
6417 }
6418 {
6419   \newcommand*\glsfmtlongpl[1]{%
6420     \glsxtrtitlelongpl{#1}}
6421 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

6422 \ifdef\txorpdfstring
6423 {
6424   \newcommand*\Glsfmtlongpl[1]{%

```

```

6425 \texorpdfstring
6426 {\Glsxtrtitlelongpl{#1}}%
6427 {\glsentrylongpl{#1}}%
6428 }
6429 }
6430 {
6431 \newcommand*{\Glsfmtlongpl}[1]{%
6432 \Glsxtrtitlelongpl{#1}}
6433 }

```

`\glsfmtfull` In-line full format.

```

6434 \ifdef\texorpdfstring
6435 {
6436 \newcommand*{\glsfmtfull}[1]{%
6437 \texorpdfstring
6438 {\glsxtrtitlefull{#1}}%
6439 {\glsxtrinlinefullformat{#1}{}}%
6440 }
6441 }
6442 {
6443 \newcommand*{\glsfmtfull}[1]{%
6444 \glsxtrtitlefull{#1}}
6445 }

```

`\Glsfmtfull` First letter converted to upper case.

```

6446 \ifdef\texorpdfstring
6447 {
6448 \newcommand*{\Glsfmtfull}[1]{%
6449 \texorpdfstring
6450 {\Glsxtrtitlefull{#1}}%
6451 {\Glsxtrinlinefullformat{#1}{}}%
6452 }
6453 }
6454 {
6455 \newcommand*{\Glsfmtfull}[1]{%
6456 \Glsxtrtitlefull{#1}}
6457 }

```

`\glsfmtfullpl` In-line full plural format.

```

6458 \ifdef\texorpdfstring
6459 {
6460 \newcommand*{\glsfmtfullpl}[1]{%
6461 \texorpdfstring
6462 {\glsxtrtitlefullpl{#1}}%
6463 {\glsxtrinlinefullplformat{#1}{}}%
6464 }
6465 }
6466 {
6467 \newcommand*{\glsfmtfullpl}[1]{%

```

```

6468 \glsxtrtitlefullpl{#1}}
6469 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

6470 \ifdef\txorpdfstring
6471 {
6472   \newcommand*{\Glsfmtfullpl}[1]{%
6473     \txorpdfstring
6474     {\Glsxtrtitlefullpl{#1}}%
6475     {\Glsxtrinlinefullplformat{#1}{}}%
6476   }
6477 }
6478 {
6479   \newcommand*{\Glsfmtfullpl}[1]{%
6480     \Glsxtrtitlefullpl{#1}}
6481 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

6482 \newcommand*{\RequireGlossariesExtraLang}[1]{%
6483   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
6484 }

```

`sariesExtraLang`

```

6485 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
6486   \ProvidesFile{glossariesxtr-#1.ldf}%
6487 }

```

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.)

```

6488 \@ifpackageloaded{tracklang}
6489 {%
6490   \AnyTrackedLanguages
6491   {%
6492     \ForEachTrackedDialect{\this@dialect}{%
6493       \IfTrackedLanguageFileExists{\this@dialect}%
6494       {glossariesxtr-}% prefix
6495       {.ldf}%
6496     }%
6497     \RequireGlossariesExtraLang{\CurrentTrackedTag}%
6498   }%
6499 }%

```

```

6500      }%
6501      }%
6502  }%
6503  {}%
6504 }
6505 {}

    Load glossaries-extra-stylemods if required.
6506 \@glsxtr@redefstyles
    and set the style:
6507 \@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:

```
6508 \NeedsTeXFormat{LaTeX2e}
6509 \ProvidesPackage{glossaries-extra-stylemods}[2017/01/19 v1.11 (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`

```
6510 \newcommand*{\@glxtr@loadstyles}{%
6511 \DeclareOption*{%
6512   \IfFileExists{glossary-\CurrentOption.sty}%
6513   {\eappto\@glxtr@loadstyles{%
6514     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
6515   {\PackageError{glossaries-extra-styles}%
6516     {Unknown option '\CurrentOption'}{}}
6517 }
```

Process the package options:

```
6518 \ProcessOptions
```

Load the required packages:

```
6519 \@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`

```
6520 \providecommand{\renewglossarystyle}[2]{%
6521   \ifcsundef{@glstyle@#1}%
6522   {%
6523     \PackageError{glossaries}{Glossary style '#1' isn't already defined}{}%

```

```

6524 }%
6525 {%
6526   \csdef{@glsstyle@#1}{#2}%
6527 }%
6528 }

```

2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying.

```

6529 \ifdefined{@glsstyle@listdotted}
6530 {%
6531   \renewglossarystyle{listdotted}{%
6532     \setglossarystyle{list}%
6533     \renewcommand*{\glossentry}[2]{%
6534       \item[]\makebox[\glslistdottedwidth][l]{%
6535         \glsentryitem{##1}%
6536         \glstarget{##1}{\glossentryname{##1}}%
6537         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
6538         \glossentrydesc{##1}\glspostdescription}%
6539     \renewcommand*{\subglossentry}[3]{%
6540       \item[]\makebox[\glslistdottedwidth][l]{%
6541         \glssubentryitem{##2}%
6542         \glstarget{##2}{\glossentryname{##2}}%
6543         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
6544         \glossentrydesc{##2}\glspostdescription}%
6545   }
6546 }
6547 {}

```

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.

```

6548 \ifcsdef{@glsstyle@long3col}
6549 {%
6550   \renewglossarystyle{long3col}{%
6551     \renewenvironment{theglossary}%
6552       {\begin{longtable}{lp{\glsdescwidth}p{\glspagerlistwidth}}}%
6553       {\end{longtable}}%
6554     \renewcommand*{\glossaryheader}{}%
6555     \renewcommand*{\glsgroupheading}[1]{}%
6556     \renewcommand{\glossentry}[2]{%
6557       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6558       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline

```

```

6559 }%
6560 \renewcommand{\subglossentry}[3]{%
6561     &
6562     \glssubentryitem{##2}%
6563     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6564     ##3\tabularnewline
6565 }%
6566 \renewcommand*\{ \glsgroupskip }{%
6567     \ifglsgnoglobskip\else & &\tabularnewline\fi}%
6568 }
6569 }
6570 {}

```

Four column style:

```

6571 \ifcsdef{@glsstyle@long4col}
6572 {%
6573     \renewglossarystyle{long4col}{%
6574         \renewenvironment{theglossary}%
6575             {\begin{longtable}{llll}}%
6576             {\end{longtable}}%
6577         \renewcommand*\{ \glossaryheader }{%
6578             \renewcommand*\{ \glsgroupheading }[1]{%
6579                 \renewcommand{\glossentry}[2]{%
6580                     \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6581                     \glossentrydesc{##1}\glspostdescription &
6582                     \glossentrysymbol{##1} &
6583                     ##2\tabularnewline
6584                 }%
6585                 \renewcommand{\subglossentry}[3]{%
6586                     &
6587                     \glssubentryitem{##2}%
6588                     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6589                     \glossentrysymbol{##2} & ##3\tabularnewline
6590                 }%
6591                 \renewcommand*\{ \glsgroupskip }{%
6592                     \ifglsgnoglobskip\else & &\tabularnewline\fi}%
6593             }
6594         }
6595     }

```

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.

```

6596 \ifcsdef{@glsstyle@longragged3col}
6597 {%
6598     \renewglossarystyle{longragged3col}{%

```

```

6599 \renewenvironment{theglossary}%
6600     {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}%
6601      >{\raggedright}p{\glspagelistwidth}}}%
6602     {\end{longtable}}}%
6603 \renewcommand*{\glossaryheader}{}%
6604 \renewcommand*{\glsgroupheading}[1]{}%
6605 \renewcommand{\glossentry}[2]{%
6606     \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6607     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
6608 }%
6609 \renewcommand{\subglossentry}[3]{%
6610     &
6611     \glssubentryitem{##2}%
6612     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6613     ##3\tabularnewline
6614 }%
6615 \renewcommand*{\glsgroupskip}{%
6616     \ifglsgnoglobskip\else & &\tabularnewline\fi}%
6617 }
6618 }
6619 {}

```

Four column style:

```

6620 \ifcsdef{@glstyle@altlongragged4col}
6621 {%
6622     \renewglossarystyle{altlongragged4col}{%
6623         \renewenvironment{theglossary}%
6624             {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}1%
6625              >{\raggedright}p{\glspagelistwidth}}}%
6626             {\end{longtable}}}%
6627         \renewcommand*{\glossaryheader}{}%
6628         \renewcommand*{\glsgroupheading}[1]{}%
6629         \renewcommand{\glossentry}[2]{%
6630             \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6631             \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
6632             ##2\tabularnewline
6633         }%
6634         \renewcommand{\subglossentry}[3]{%
6635             &
6636             \glssubentryitem{##2}%
6637             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6638             \glossentrysymbol{##2} & ##3\tabularnewline
6639         }%
6640         \renewcommand*{\glsgroupskip}{%
6641             \ifglsgnoglobskip\else & & &\tabularnewline\fi}%
6642     }
6643 }
6644 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment, but not the two column styles.

```
6645 \ifcsdef{@glsstyle@super3col}
6646 {%
6647   \renewglossarystyle{super3col}{%
6648     \renewenvironment{theglossary}%
6649       {\tablehead{}\tabletail{}}%
6650       \begin{supertabular}{\lp{@glsdescwidth}p{@glspagelistwidth}}}%
6651       {\end{supertabular}}%
6652     \renewcommand*{\glossaryheader}{}%
6653     \renewcommand*{\glsgroupheading}[1]{}%
6654     \renewcommand{\glossentry}[2]{%
6655       \glstryitem{##1}\glstarget{##1}{\glsentryname{##1}} &
6656       \glsentrydesc{##1}\glspostdescription & ##2\tabularnewline
6657     }%
6658     \renewcommand{\subglossentry}[3]{%
6659       &
6660       \glssubentryitem{##2}%
6661       \glstarget{##2}{\strut}\glsentrydesc{##2}\glspostdescription &
6662       ##3\tabularnewline
6663     }%
6664     \renewcommand*{\glsgroupskip}{%
6665       \ifglsnogroupskip\else & &\tabularnewline\fi}%
6666   }
6667 }
6668 {}
```

Four column styles:

```
6669 \ifcsdef{@glsstyle@super4col}
6670 {%
6671   \renewglossarystyle{super4col}{%
6672     \renewenvironment{theglossary}%
6673       {\tablehead{}\tabletail{}}%
6674       \begin{supertabular}{\llll}}%
6675       {\end{supertabular}}%
6676     \renewcommand*{\glossaryheader}{}%
6677     \renewcommand*{\glsgroupheading}[1]{}%
6678     \renewcommand{\glossentry}[2]{%
6679       \glstryitem{##1}\glstarget{##1}{\glsentryname{##1}} &
6680       \glsentrydesc{##1}\glspostdescription &
6681       \glsentrysymbol{##1} & ##2\tabularnewline
6682     }%
6683     \renewcommand{\subglossentry}[3]{%
6684       &
6685       \glssubentryitem{##2}%
6686       \glstarget{##2}{\strut}\glsentrydesc{##2}\glspostdescription &
6687       \glsentrysymbol{##2} & ##3\tabularnewline
6688     }%
6689     \renewcommand*{\glsgroupskip}{%
6690       \ifglsnogroupskip\else & & &\tabularnewline\fi}%
6691   }
6692 }
```

```

6690     \ifglsnogroupskip\else & & \tabularnewline\fi}%
6691   }
6692 }
6693 {}

```

2.6 Super Ragged Styles

The three and four column styles require adjustment, but not the two column styles.

```

6694 \ifcsdef{@glsstyle@superragged3col}
6695 {%
6696   \renewglossarystyle{superragged3col}{%
6697     \renewenvironment{theglossary}%
6698       {\tablehead{}\tabletail}%
6699       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%
6700         >{\raggedright}p{\glspagelistwidth}}}%
6701       {\end{supertabular}}}%
6702   \renewcommand*{\glossaryheader}{}%
6703   \renewcommand*{\glsgroupheading}[1]{}%
6704   \renewcommand{\glossentry}[2]{%
6705     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6706     \glossentrydesc{##1}\glspostdescription &
6707     ##2\tabularnewline
6708   }%
6709   \renewcommand{\subglossentry}[3]{%
6710     &
6711     \glssubentryitem{##2}%
6712     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6713     ##3\tabularnewline
6714   }%
6715   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
6716     &\tabularnewline\fi}%
6717 }
6718 }
6719 {}

```

Four columns:

```

6720 \ifcsdef{@glsstyle@altsuperragged4col}
6721 {%
6722   \renewglossarystyle{altsuperragged4col}{%
6723     \renewenvironment{theglossary}%
6724       {\tablehead{}\tabletail}%
6725       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}l%
6726         >{\raggedright}p{\glspagelistwidth}}}%
6727       {\end{supertabular}}}%
6728   \renewcommand*{\glossaryheader}{}%
6729   \renewcommand{\glossentry}[2]{%
6730     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6731     \glossentrydesc{##1}\glspostdescription &
6732     \glossentrysymbol{##1} & ##2\tabularnewline

```

```

6733 }%
6734 \renewcommand{\subglossentry}[3]{%
6735     &
6736     \glssubentryitem{##2}%
6737     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6738     \glossentrysymbol{##2} & ##3\tabularnewline
6739 }%
6740 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & &
6741     &\tabularnewline\fi}%
6742 }
6743 }
6744 {}

```

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`.

```

6745 \ifdef{\@glsstyle@inline}
6746 {%
6747     \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
        Just use \glsxtrpostdescription instead of \glspostdescription.
6748     \renewcommand*{\glsinlinedescformat}[3]{%
6749         \space#1\glsxtrpostdescription}
6750     \renewcommand*{\glsinlinesubdescformat}[3]{%
6751         #1\glsxtrpostdescription}
6752 }
6753 {}

```

2.8 Tree Styles

The `alttree` style is redefined to make it easier to made minor adjustments.

```

6754 \ifdef{\@glsstyle@alttree}
6755 {%

```

Only redefine this style if it's already been defined.

mbolDescLocation

```
\glxtraltrtreeSymbolDescLocation{\langle label \rangle}{\langle location list \rangle}
```

Layout the symbol, description and location for top-level entries.

```

6756 \newcommand{\glxtraltrtreeSymbolDescLocation}[2]{%
6757     {%
6758         \let\par\glsxtrAltTreePar

```

```

6759      \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
6760      \glossentrydesc{#1}\glspostdescription \space #2\par
6761  }%
6762  }

```

`\trAltTreeIndent` Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

6763  \newlength\glxstrAltTreeIndent

```

`\lsxtrAltTreePar` Multi-paragraph descriptions need to keep the hanging indent.

```

6764  \newcommand{\glxstrAltTreePar}{%
6765    @@par
6766    \glxstrAltTreeSetHangIndent
6767    \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
6768  }

```

`\symbolDescLocation` `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

6769  \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
6770    \glxstralttreeSymbolDescLocation{#2}{#3}%
6771  }

```

`\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.

```

6772  \newlength\glxstrtreeindent

```

`\glxstralttreeInit` User-level initialisation for the alttree style.

```

6773  \newcommand*{\glxstralttreeInit}{%
6774    \settowidth{\glxstrtreeindent}{\glstreenamfmt{\glsgetwidestname\space}}%
6775    \glxstrAltTreeIndent=\parindent
6776  }

```

`\eglissetwidest` The original `\glissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

6777  \newcommand*{\eglissetwidest}[2][0]{%
6778    \protected@csedef{@glswidestname\romannumeral#1}{#2}%
6779  }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```

6780  \newcommand*{\xglissetwidest}[2][0]{%
6781    \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
6782  }

```

`\lsgetwidestname` Provide a user-level macro to obtain the widest top-level name.

```

6783  \newcommand*{\glsggetwidestname}{\@glswidestname}

```

`etwidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```
6784 \newcommand*\glsgetwidestsubname}[1]{%
6785   \ifcsundef{@glswidestname\romannumeral#1}%
6786     {\@glswidestname}%
6787     {\csuse{@glswidestname\romannumeral#1}}%
6788 }
```

`estTopLevelName` CamelCase is easier for long command names. Provide a CamelCase synonym of `\glsfindwidesttoplevelname`

```
6789 \let\glsFindWidestTopLevelName\glsfindwidesttoplevelname
```

`sedTopLevelName` 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.

```
6790 \newrobustcmd*\glsFindWidestUsedTopLevelName}[1][\@glo@types]{%
6791   \dimen@=0pt\relax
6792   \gls@tmplen=0pt\relax
6793   \foralllglossaries[#1]{\@gls@type}%
6794   {%
6795     \forglsentries[\@gls@type]{\@glo@label}%
6796     {%
6797       \ifglsused{\@glo@label}%
6798       {%
6799         \ifglshasparent{\@glo@label}%
6800         {}%
6801         {%
6802           \settowidth{\dimen@}%
6803             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
6804           \ifdim\dimen@>\gls@tmplen
6805             \gls@tmplen=\dimen@
6806             \eglssetwidest{\glsentryname{\@glo@label}}%
6807           \fi
6808         }%
6809       }%
6810     }%
6811   }%
6812 }%
6813 }
```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```
6814 \newrobustcmd*\glsFindWidestUsedAnyName}[1][\@glo@types]{%
6815   \dimen@=0pt\relax
6816   \gls@tmplen=0pt\relax
6817   \foralllglossaries[#1]{\@gls@type}%
6818   {%
6819     \forglsentries[\@gls@type]{\@glo@label}%
6820     {%
```

```

6821     \ifglused{\@glo@label}%
6822     {%
6823         \settowidth{\dimen@}%
6824         {\glstreenamefmt{\glentryname{\@glo@label}}}%
6825         \ifdim\dimen@>\gls@tmplen
6826             \gls@tmplen=\dimen@
6827             \eglssetwidest{\glentryname{\@glo@label}}%
6828         \fi
6829     }%
6830 }%
6831 }%
6832 }%
6833 }

```

FindWidestAnyName Like the above but doesn't check if the entry has been used.

```

6834 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
6835     \dimen@=0pt\relax
6836     \gls@tmplen=0pt\relax
6837     \forallglossaries[#1]{\@gls@type}%
6838     {%
6839         \forallglentries[\@gls@type]{\@glo@label}%
6840         {%
6841             \settowidth{\dimen@}%
6842             {\glstreenamefmt{\glentryname{\@glo@label}}}%
6843             \ifdim\dimen@>\gls@tmplen
6844                 \gls@tmplen=\dimen@
6845                 \eglssetwidest{\glentryname{\@glo@label}}%
6846             \fi
6847         }%
6848     }%
6849 }

```

FindWidestUsedLevelTwo This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

6850 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
6851     \dimen@=0pt\relax
6852     \dimen@i=0pt\relax
6853     \dimen@ii=0pt\relax
6854     \forallglossaries[#1]{\@gls@type}%
6855     {%
6856         \forallglentries[\@gls@type]{\@glo@label}%
6857         {%
6858             \ifglused{\@glo@label}%
6859             {%
6860                 \ifglshasparent{\@glo@label}%
6861                 {%
6862                     \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
6863                     \ifglshasparent{\@glo@parent}%
6864                 }%

```

```

6865         \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
6866         \ifglshasparent{\@glo@parent}%
6867         {%
6868         {%
6869             \settowidth{\gls@tmplen}%
6870             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6871             \ifdim\gls@tmplen>\dimen@ii
6872             \dimen@ii=\gls@tmplen
6873             \eglssetwidest[2]{\glsentryname{\@glo@label}}%
6874             \fi
6875         }%
6876     }%
6877     {%
6878         \settowidth{\gls@tmplen}%
6879         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6880         \ifdim\gls@tmplen>\dimen@i
6881         \dimen@i=\gls@tmplen
6882         \eglssetwidest[1]{\glsentryname{\@glo@label}}%
6883         \fi
6884     }%
6885 }%
6886 {%
6887     \settowidth{\gls@tmplen}%
6888     {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6889     \ifdim\gls@tmplen>\dimen@
6890     \dimen@=\gls@tmplen
6891     \eglssetwidest{\glsentryname{\@glo@label}}%
6892     \fi
6893 }%
6894 }%
6895 {}%
6896 }%
6897 }%
6898 }

```

`dWidestLevelTwo` This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

6899 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
6900     \dimen@=0pt\relax
6901     \dimen@i=0pt\relax
6902     \dimen@ii=0pt\relax
6903     \foralllglossaries[#1]{\@gls@type}%
6904     {%
6905         \forglsentries[\@gls@type]{\@glo@label}%
6906         {%
6907             \ifglshasparent{\@glo@label}%
6908             {%
6909                 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
6910                 \ifglshasparent{\@glo@parent}%
6911                 {%

```

```

6912         \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
6913         \ifglshasparent{\@glo@parent}%
6914         {}%
6915         {%
6916             \settowidth{\gls@tmplen}%
6917                 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6918             \ifdim\gls@tmplen>\dimen@ii
6919                 \dimen@ii=\gls@tmplen
6920                 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
6921             \fi
6922         }%
6923     }%
6924     {%
6925         \settowidth{\gls@tmplen}%
6926             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6927         \ifdim\gls@tmplen>\dimen@i
6928             \dimen@i=\gls@tmplen
6929             \eglssetwidest[1]{\glsentryname{\@glo@label}}%
6930         \fi
6931     }%
6932 }%
6933 {%
6934     \settowidth{\gls@tmplen}%
6935         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6936     \ifdim\gls@tmplen>\dimen@
6937         \dimen@=\gls@tmplen
6938         \eglssetwidest{\glsentryname{\@glo@label}}%
6939     \fi
6940 }%
6941 }%
6942 }%
6943 }

```

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.

```

6944 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
6945     \dimen@=0pt\relax
6946     \gls@tmplen=0pt\relax
6947     #2=0pt\relax
6948     \forallglossaries[#1]{\@gls@type}%
6949     {%
6950         \forglsentries[\@gls@type]{\@glo@label}%
6951         {%
6952             \ifglused{\@glo@label}%
6953             {%
6954                 \settowidth{\dimen@}%
6955                     {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6956                 \ifdim\dimen@>\gls@tmplen
6957                     \gls@tmplen=\dimen@

```

```

6958         \eglssetwidest{\glsentryname{\@glo@label}}}%
6959     \fi
6960     \settowidth{\dimen@}%
6961     {\glsentrysymbol{\@glo@label}}}%
6962     \ifdim\dimen@>#2\relax
6963         #2=\dimen@
6964     \fi
6965 }%
6966 {}%
6967 }%
6968 }%
6969 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

6970 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
6971     \dimen@=0pt\relax
6972     \gls@tmplen=0pt\relax
6973     #2=0pt\relax
6974     \forallglossaries[#1]{\@gls@type}%
6975     {%
6976         \forglsentries[\@gls@type]{\@glo@label}%
6977         {%
6978             \settowidth{\dimen@}%
6979             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
6980             \ifdim\dimen@>\gls@tmplen
6981                 \gls@tmplen=\dimen@
6982                 \eglssetwidest{\glsentryname{\@glo@label}}}%
6983             \fi
6984             \settowidth{\dimen@}%
6985             {\glsentrysymbol{\@glo@label}}}%
6986             \ifdim\dimen@>#2\relax
6987                 #2=\dimen@
6988             \fi
6989         }%
6990     }%
6991 }

```

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.

```

6992 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
6993     \dimen@=0pt\relax
6994     \gls@tmplen=0pt\relax
6995     #2=0pt\relax
6996     #3=0pt\relax
6997     \forallglossaries[#1]{\@gls@type}%
6998     {%
6999         \forglsentries[\@gls@type]{\@glo@label}%

```

```

7000    {%
7001        \ifglsused{\@glo@label}%
7002    {%
7003        \settowidth{\dimen@}%
7004            {\glstreenamefmt{\glentryname{\@glo@label}}}%
7005        \ifdim\dimen@>\gls@tmplen
7006            \gls@tmplen=\dimen@
7007            \eglssetwidest{\glentryname{\@glo@label}}%
7008        \fi
7009        \settowidth{\dimen@}%
7010            {\glentrysymbol{\@glo@label}}%
7011        \ifdim\dimen@>#2\relax
7012            #2=\dimen@
7013        \fi
7014        \settowidth{\dimen@}%
7015            {\GlsXtrFormatLocationList{\glentrynumberlist{\@glo@label}}}%
7016        \ifdim\dimen@>#3\relax
7017            #3=\dimen@
7018        \fi
7019    }%
7020    {%
7021    }%
7022    }%
7023 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

7024 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
7025     \dimen@=0pt\relax
7026     \gls@tmplen=0pt\relax
7027     #2=0pt\relax
7028     #3=0pt\relax
7029     \forallglossaries[#1]{\@gls@type}%
7030 {%
7031     \forglsentries[\@gls@type]{\@glo@label}%
7032     {%
7033         \settowidth{\dimen@}%
7034             {\glstreenamefmt{\glentryname{\@glo@label}}}%
7035         \ifdim\dimen@>\gls@tmplen
7036             \gls@tmplen=\dimen@
7037             \eglssetwidest{\glentryname{\@glo@label}}%
7038         \fi
7039         \settowidth{\dimen@}%
7040             {\glentrysymbol{\@glo@label}}%
7041         \ifdim\dimen@>#2\relax
7042             #2=\dimen@
7043         \fi
7044         \settowidth{\dimen@}%
7045             {\GlsXtrFormatLocationList{\glentrynumberlist{\@glo@label}}}%
7046         \ifdim\dimen@>#3\relax

```

```

7047         #3=\dimen@
7048     \fi
7049 }%
7050 }%
7051 }

```

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.

```

7052 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
7053     \dimen@=0pt\relax
7054     \gls@tmplen=0pt\relax
7055     #2=0pt\relax
7056     \forallglossaries[#1]{\@gls@type}%
7057     {%
7058         \forglseentries[\@gls@type]{\@glo@label}%
7059         {%
7060             \ifglssused{\@glo@label}%
7061             {%
7062                 \settowidth{\dimen@}%
7063                 {\glstreenamfmt{\glseentryname{\@glo@label}}}%
7064                 \ifdim\dimen@>\gls@tmplen
7065                     \gls@tmplen=\dimen@
7066                     \eglsssetwidest{\glseentryname{\@glo@label}}%
7067                 \fi
7068                 \settowidth{\dimen@}%
7069                 {\GlsXtrFormatLocationList{\glseentrynumberlist{\@glo@label}}}%
7070                 \ifdim\dimen@>#2\relax
7071                     #2=\dimen@
7072                 \fi
7073             }%
7074         }%
7075     }%
7076 }%
7077 }

```

AnyNameLocation Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

7078 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
7079     \dimen@=0pt\relax
7080     \gls@tmplen=0pt\relax
7081     #2=0pt\relax
7082     \forallglossaries[#1]{\@gls@type}%
7083     {%
7084         \forglseentries[\@gls@type]{\@glo@label}%
7085         {%
7086             \settowidth{\dimen@}%
7087             {\glstreenamfmt{\glseentryname{\@glo@label}}}%
7088             \ifdim\dimen@>\gls@tmplen
7089                 \gls@tmplen=\dimen@

```

```

7090      \eglssetwidest{\glstryname{\@glo@label}}}%
7091      \fi
7092      \settowidth{\dimen@}%
7093      {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
7094      \ifdim\dimen@>#2\relax
7095          #2=\dimen@
7096      \fi
7097  }%
7098  }%
7099  }

```

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`.

```

7100  \newcommand*{\glsxtrComputeTreeIndent}[1]{%
7101      \glstreeindent=\glsxtrtreetopindent\relax
7102  }

```

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.

```

7103  \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
7104      \ifcsundef{@glswidestname\romannumeral#1}%
7105      {%
7106          \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
7107      }%
7108      {%
7109          \settowidth{#3}{\glstreenamefmt{%
7110              \csname @glswidestname\romannumeral#1\endcsname\space}}%
7111      }%
7112  }

```

treeSetHangIndent Set `\hangindent` for top-level entries:

```

7113 \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

treeSubHangIndent Set `\hangindent` for sub-entries:

```

7114 \newcommand*{\glsxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine `alttree`:

```

7115  \renewglossarystyle{alttree}{%
7116      \renewenvironment{theglossary}%
7117      {%
7118          \glsxtralttreeInit

```

```

7119     \def\@gls@prevlevel{-1}%
7120     \mbox{}\par}%
7121     {\par}%
7122 \renewcommand*\glossaryheader{}%
7123 \renewcommand*\glsgroupheading[1]{}%
7124 \renewcommand\glossentry[2]{%
7125     \ifnum\@gls@prevlevel=0\relax
7126     \else
7127         \glstrComputeTreeIndent{##1}%
7128     \fi
7129     \parindent\glstreeindent
7130     \glstrAltTreeSetHangIndent
7131     \makebox[Opt][r]%
7132     {%
7133         \glstreenamebox{\glstreeindent}%
7134         {%
7135             \glstryitem{##1}%
7136             \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
7137         }%
7138     }%
7139     \glstralttreeSymbolDescLocation{##1}{##2}%
7140 \def\@gls@prevlevel{0}%
7141 }
7142 \renewcommand\subglossentry[3]{%
7143     \ifnum##1=1\relax
7144         \glssubentryitem{##2}%
7145     \fi
7146     \ifnum\@gls@prevlevel=##1\relax
7147     \else
7148         \glstrComputeTreeSubIndent{##1}{##2}{\@gls@tmplen}%
7149         \ifnum\@gls@prevlevel<##1\relax
7150             \setlength\glstreeindent\@gls@tmplen
7151             \addtolength\glstreeindent\parindent
7152             \parindent\glstreeindent
7153         \else
7154             \ifnum\@gls@prevlevel=0\relax
7155                 \glstrComputeTreeIndent{##2}%
7156             \else
7157                 \glstrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
7158             \fi
7159             \addtolength\parindent{-\glstreeindent}%
7160             \setlength\glstreeindent\parindent
7161         \fi
7162     \fi
7163     \glstrAltTreeSetSubHangIndent{##1}%
7164     \makebox[Opt][r]{\glstreenamebox{\@gls@tmplen}{%
7165         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
7166     \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
7167 \def\@gls@prevlevel{##1}%

```

```

7168     }%
7169     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7170   }
7171 }%
7172 {%

```

Assume the style isn't required if it hasn't already been defined.

```

7173 }

```

Reset the default style

```

7174 \ifx\@glossary@default@style\relax
7175 \else
7176   \setglossarystyle{\@glxtr@current@style}
7177 \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	37
General: Initial experimental release	4	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	37
\Glsfmtshort: new	184	\@Glstext@: added redefinition	33
\glsfmtshort: new	183	\@Glsuseri@: added redefinition	38
\Glsfmtshortpl: new	184	\@Glsuserii@: added redefinition	38
\glsfmtshortpl: new	183	\@Glsuseriii@: added redefinition	38
short: switched inline full form to short		\@Glsuseriv@: added redefinition	38
(long)	150	\@Glsuserv@: added redefinition	39
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	39
\@ACRlong: added redefinition	42	\@acrlong: added redefinition	42
\@ACRlongpl: added redefinition	43	\@acrlongpl: added redefinition	43
\@ACRshort: added redefinition	40	\@acrshort: added redefinition	39
\@ACRshortpl: added redefinition	41	\@acrshortpl: added redefinition	40
\@Acrlong: added redefinition	42	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	43	argument	30
\@Acrshort: added redefinition	40	\@glsdescplural@: added redefinition	36
\@Acrshortpl: added redefinition	41	\@glsfirst@: added redefinition	33
\@GLSdesc@: added redefinition	36	\@glsfirstplural@: added redefinition	34
\@GLSdescplural@: added redefinition	36	\@glsplural@: added redefinition	34
\@GLSfirst@: added redefinition	33	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	35	redefinition	37
\@GLSname@: added redefinition	35	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	34	new	84
\@GLSSymbol@: added redefinition	37	\@glsxtr@field@linkdefs: new	32
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	123
redefinition	37	\@print@glossary: added redefinition	81
\@GLStext@: added redefinition	32	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	38	\abbrvdefaultfont	127
\@GLSuserii@: added redefinition	38	\glsaccessdesc: new	93
\@GLSuseriii@: added redefinition	38	\glsaccessdescplural: new	93
\@GLSuseriv@: added redefinition	39	\glsaccessfirst: new	91
\@GLSuserv@: added redefinition	39	\glsaccessfirstplural: new	91
\@GLSuservi@: added redefinition	39	\Glsaccesslong: new	95
\@Glsdesc@: added redefinition	36	\glsaccesslong: new	95
\@Glsdescplural@: added redefinition	36	\glsaccessname: new	89
\@Glsfirst@: added redefinition	33	\glsaccessplural: new	90
\@Glsfirstplural@: added redefinition	35	\Glsaccessshort: new	94
\@Glsname@: added redefinition	35	\glsaccessshort: new	94
\@Glsplural@: added redefinition	34	\Glsaccessshortpl: new	95

\glsaccesssshortpl: new	95	\cGLSpl: new	65
\glsaccessssymbol: new	92	\cGLSpl@: new	65
\glsaccessssymbolplural: new	92	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	89	new	60
\glstentryfmt: added check for short ..	29	\cGLS: new	64
\glslongpltok: new	123	\cGLSformat: new	64
\glsshortpltok: new	123	\cGLSpl: new	64
\glsxtrdiscardperiod: added check		\cGLSplformat: new	65
for plural	120	\GlossariesExtraWarningNoLine:	
\GLSxtrlongpl: new	136	new	10
\Glsxtrlongpl: new	135	\glsenableentrycount: new	60
\glsxtrlongpl: new	135	\glsfirstabbrvdefaultfont: new ..	126
\glsxtrNoGlossaryWarning: new	13	\glsfirstlongdefaultfont: new ...	127
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirst: new	186
new	119	\Glsfmtfirst: new	185
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtfirstpl: new	186
new	119	\Glsfmtfirstpl: new	186
\glsxtrpostlinkendsentence: new ..	119	\Glsfmtplural: new	185
\GLSxtrshortpl: new	134	\Glsfmtplural: new	185
\Glsxtrshortpl: new	134	\Glsfmtshort: changed to use	
\glsxtrshortpl: new	133	\Glsxtrtitleshort	184
short-long-desc: fixed name to use		renamed from \Glstentryfmtshort ..	184
\glslabeltok	145	\Glsfmtshort: changed to use	
\newabbreviation: fixed family name in		\glsxtrtitleshort	183
\setkeys	123	renamed from \glstentryfmtshort ..	183
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	143	\Glsxtrtitleshortpl	184
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl	184
redefinition of \acronymtype	11	\Glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	183
\Glsxtrshort	184	renamed from	
\Glsfmtshort: changed to use		\glstentryfmtshortpl	183
\glsxtrshort	183	\Glsfmttext: new	185
\Glsfmtshortpl: changed to use		\Glsfmttext: new	184
\glsxtrshortpl	184	\glshasattribute: new	101
\Glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	100
\glsxtrshortpl	183	\GlsXtrEnableEntryCounting: new ..	59
\glsxtrifemptyglossary: new	16	\glsxtrifcounttrigger: new	62
\glsxtrnewnumber: added extra		\glsxtrscfont: new	154
argument	104	\glsxtrscsuffix: new	155
\glsxtrnewsymbol: added extra		\glsxtrsmfont: new	158
argument	104	\glsxtrsmsuffix: new	159
\MakeAcronymsAbbreviations: set the		short-em: new	165
default type to \acronymtype	73	short-em-desc: new	166
\newterm: fixed name argument	103	short-em-footnote: new	167
0.5 (2015-12-07)		short-em-long: new	164
\cGLS: new	64	short-em-long-desc: new	164
\cGLS@: new	64	short-em-postfootnote: new	168

short-sc-footnote: new	158	\Glsxtrheadtext: now uses headuc	
short-sc-postfootnote: new	158	attribute	177
short-sm: new	160	\glxtrheadtext: now uses headuc	
short-sm-desc: new	160	attribute	176
short-sm-footnote: new	161	short-long: switch off regular attribute	
short-sm-long: new	159	if set	144
short-sm-long-desc: new	159	short-long-desc: switch off regular	
short-sm-postfootnote: new	161	attribute if set	145
long-noshort-em: new	166	long-short: switch off regular attribute	
long-noshort-em-desc: new	167	if set	142
long-noshort-sm: new	160	long-short-desc: switch off regular	
long-noshort-sm-desc: new	161	attribute if set	144
long-short-em: new	162	footnote: switch off regular attribute if	
long-short-em-desc: new	162	set	146
long-short-sm: new	159	postfootnote: switch off regular	
long-short-sm-desc: new	159	attribute if set	148
0.5.1 (2015-12-02)		0.5.2 (2015-12-08)	
\Glsaccessstext: new	90	\@GLSdesc@: added accessibility support	36
0.5.1 (2015-12-07)		\@GLSdescplural@: added accessibility	
\@glxtr@doaccsupp: new	13	support	36
General: removed \ifglxtruseuhead	175	\@GLSfirst@: added accessibility	
\Glsaccessdesc: new	93	support	33
\Glsaccessdescplural: new	94	\@GLSfirstplural@: added accessibility	
\Glsaccessfirst: new	91	support	35
\Glsaccessfirstplural: new	91	\@GLSname@: added accessibility support	35
\Glsaccessname: new	89	\@GLSplural@: added accessibility	
\Glsaccessplural: new	90	support	34
\Glsaccesssymbol: new	92	\@GLSsymbol@: added accessibility	
\Glsaccesssymbolplural: new	92	support	37
\Glsxtrheadfirst: now uses headuc		\@GLSsymbolplural@: added	
attribute	178	accessibility support	37
\glxtrheadfirst: now uses headuc		\@GLStext@: added accessibility support	32
attribute	178	\@GLSdesc@: added accessibility support	36
\Glsxtrheadfirstplural: now uses		\@GLSdescplural@: added accessibility	
headuc attribute	179	support	36
\glxtrheadfirstplural: now uses		\@GLSfirst@: added accessibility	
headuc attribute	179	support	33
\Glsxtrheadplural: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	178	support	35
\glxtrheadplural: now uses headuc		\@GLSname@: add accessibility support ..	35
attribute	177	\@GLSplural@: added accessibility	
\Glsxtrheadshort: now uses headuc		support	34
attribute	176	\@GLSsymbol@: added accessibility	
\glxtrheadshort: now uses headuc		support	37
attribute	175	\@GLSsymbolplural@: added	
\Glsxtrheadshortpl: now uses headuc		accessibility support	37
attribute	176	\@GLStext@: added accessibility support	33
\glxtrheadshortpl: now uses headuc		\@GLSdesc@: added accessibility support	36
attribute	175		

\@glsdescplural@: added accessibility support	36	\glsxtrnewabbrevpresetkeyhook: new	125
\@glsfirst@: added accessibility support	33	\glsxtrtagfont: new	117
\@glsfirstplural@: added accessibility support	34	\KV@printgloss@nonumberlist: added	29
\@glsname@: added accessibility support	35	\mfu@checkword@do: added	117
\@glsplural@: added accessibility support	34	\setabbreviationstyle: added check for post-definition style switch	139
\@glsymbol@: added accessibility support	37	0.5.3 (2015-12-09)	
\@glsymbolplural@: added accessibility support	37	\@glsxtr@autoindex@at: new	113
\@glstext@: added accessibility support	32	\@glsxtr@autoindex@encap: new ...	113
\@glsxtr@activate@initialtagging: new	117	\@glsxtr@autoindex@esc: new	114
\@glsxtr@do@titlecaps@warn: new	117	\@glsxtr@autoindex@level: new ...	114
\@glsxtr@tag: new	117	\@glsxtr@autoindex@setname: new	112
General: fixed typo in glossaries-accsupp and tidied up code to use just one		\@glsxtr@doabbreviationsdef: new	10
\@ifpackageloaded	89	General: removed	
removed \glsxtrabbrvfmt	137	\GlsXtrNoGlsWarningNoAutoMakeMain	83
\glossaryentrynumbers: added	26	\glsdescwidth: added	26
\Glossentrydesc: added	115	\glspagelistwidth: added	26
\Glossentryname: added	109	\glsxtrdoautoindexname: new	111
\Glossentrysymbol: added	116	\glsxtrpostnamehook: new	111
\glossentrysymbol: added	115	\if@glsxtr@format@override: new	111
\GLSaccessdesc: new	93, 98	\ProvidesGlossariesExtraLang: new	189
\GLSaccessdescplural: new	94, 98	\RequireGlossariesExtraLang: new	189
\GLSaccessfirst: new	91, 97	0.5.4 (2015-12-15)	
\GLSaccessfirstplural: new	91, 97	\@newglossaryentry@defunitcounters: new	65
\GLSaccesslong: new	95, 99	\@GLSxtr@p@acrlong@: new	53
\GLSaccesslongpl: new	96, 99	\@GLSxtr@p@acrlongpl@: new	53
\Glsaccesslongpl: new	96	\@GLSxtr@p@acrshort@: new	52
\glsaccesslongpl: new	96	\@GLSxtr@p@acrshortpl@: new	53
\GLSaccessname: new	89, 96	\@GLSxtr@p@long@: new	52
\GLSaccessplural: new	90, 97	\@GLSxtr@p@longpl@: new	52
\GLSaccessshort: new	94, 98	\@GLSxtr@p@plural@: new	51
\GLSaccessshortpl: new	95, 99	\@GLSxtr@p@short@: new	51
\GLSaccesssymbol: new	92, 97	\@GLSxtr@p@shortpl@: new	52
\GLSaccesssymbolplural: new ...	93, 98	\@GLSxtr@p@text@: new	51
\GLSaccessstext: new	90, 97	\@GlsXtrEnableOnTheFly: new	22
\glsentryfmt: moved		\@Glsxtr: new	23
\glssetabbrvfmt from		\@Glsxtr@p@acrlong@: new	53
\glsxtrabbrvfmt to here	29	\@Glsxtr@p@acrlongpl@: new	53
\GlsXtrEnableInitialTagging: new	116	\@Glsxtr@p@acrshort@: new	52
\glsxtrfieldtitlecase: new	105	\@Glsxtr@p@acrshortpl@: new	53
\GlsXtrFormatLocationList: new ...	27	\@Glsxtr@p@long@: new	52
		\@Glsxtr@p@longpl@: new	52
		\@Glsxtr@p@plural@: new	51
		\@Glsxtr@p@short@: new	51
		\@Glsxtr@p@shortpl@: new	52
		\@Glsxtr@p@text@: new	51

\@Glsxtrpl: new	24	\glxtr: new	22
\@alt@glshyp@opt: new	48	\glxtrcat: new	22
\@glscalt@hyp@opt: new	48	\glxtrdowrglossaryhook: new	48
\@glscalt@hyp@opt@char: new	48	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	48	new	71
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new	22
new	66	\Glsxtrpl: new	24
\@glsc@local@increment@currunitcount:		\glxtrpl: new	23
new	67	\glxtrpostlocalreset: new	59
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new	59
new	46	\glxtrpostreset: new	59
\@glxtr: new	23	\glxtrpostunset: new	59
\@glxtr@addunitcounter: new	66	\glxtrprotectlinks: new	50
\@glxtr@currunitcount: new	67	\GlsXtrSetAltModifier: new	48
\@glxtr@ifunitcounter: new	66	\GlsXtrSetDefaultGlsOpts: new	47
\@glxtr@p@acrlong@: new	53	\glxtrstarflywarn: new	22
\@glxtr@p@acrlongpl@: new	53	\GlsXtrWarning: new	24
\@glxtr@p@acrshort@: new	52	\MakeAcronymsAbbreviations: now	
\@glxtr@p@acrshortpl@: new	53	disables \setacronymstyle	73
\@glxtr@p@long@: new	52	1.0 (2016-01-24)	
\@glxtr@p@longpl@: new	52	\@glxtr@autoindexcrossrefs: new .	10
\@glxtr@p@plural@: new	51	\@glxtr@idx@displaynumberlist:	
\@glxtr@p@short@: new	51	new	78
\@glxtr@p@shortpl@: new	51	\@glxtr@idx@entrynumberlist: new	80
\@glxtr@p@text@: new	51	\@glxtr@noidx@displaynumberlist:	
\@glxtr@prevunitcount: new	67	new	78
\@glxtr@setentryunitcountunsetattr:		\@glxtr@noidx@entrynumberlist:	
new	71	new	79
\@glxtr@unitcountlist: new	66	\@glxtr@noidx@numberlistloop:	
\@glxtrpl: new	23	new	79
\@newglossaryentryposthook: added		\@glxtr@reg@glosslist: new	74
empty see value if not set and added		\makeglossaries: new	74
‘see’ to field key map	19	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	22	\glxtrdiscardperiod: added check	
\cGlsformat: added	65	for first use	120
\cGlsformat: added	65	short-desc: fixed typo in	
\cGlsplformat: added	65	\glxtrinlinefullformat and	
\cGlsplformat: added	65	added missing second argument ...	151
\glsdisablehyper: added	49	1.02 (2016-04-25)	
\glsdohyperlink: added	49	\@glxtr@current@style: new	25
\glsdonohyperlink: added	50	\Glsfmtfull: new	188
\glsenableentryunitcount: new	68	\glsfmtfull: new	188
\glshasattribute: added check for		\Glsfmtfullpl: new	189
entry’s existence	101	\glsfmtfullpl: new	188
\glusifattribute: added check for		\Glsfmtlong: new	187
entry’s existence	101	\glsfmtlong: new	187
\glspostlinkhook: added existence		\Glsfmtlongpl: new	187
check	118	\glsfmtlongpl: new	187
\Glsxtr: new	23	\Glsxtrheadfull: new	182

\glxtrheadfull: new	181	\@GLSplural@: set abbreviation and regular format	34
\Glsxtrheadfullpl: new	183	\@GLSsymbol@: set regular format	37
\glxtrheadfullpl: new	182	\@GLSsymbolplural@: set regular format	37
\glxtrheadlong: new	180	\@GLStext@: set abbreviation and regular format	32
\glxtrheadlongpl: new	181	\@GLSuseri@: set regular format	38
\glxtrheadlongpl: new	180	\@GLSuserii@: set regular format	38
\Glsxtrtitlefull: new	182	\@GLSuseriii@: set regular format	38
\glxtrtitlefull: new	182	\@GLSuseriv@: set regular format	39
\Glsxtrtitlefullpl: new	183	\@GLSuserv@: set regular format	39
\glxtrtitlefullpl: new	182	\@GLSuservi@: set regular format	39
\Glsxtrtitlelong: new	181	\@Glsdesc@: set abbreviation and regular format	36
\glxtrtitlelong: new	180	\@Glsdescplural@: set abbreviation and regular format	36
\Glsxtrtitlelongpl: new	181	\@Glsfirst@: set abbreviation and regular format	33
\glxtrtitlelongpl: new	180	\@Glsfirstplural@: set abbreviation and regular format	35
\ifglxtrinsetinside: new	142	\@Glsname@: set abbreviation and regular format	35
postfootnote: added redef of \glxtrsetupfulldefs	148	\@Glsplural@: set abbreviation and regular format	34
stylemods: new	14	\@Glsymbol@: set regular format	37
1.03 (2016-04-27)		\@Glsymbolplural@: set regular format	37
\@GLSfirstplural@: bug fix: misspelt cs name	35	\@Glstext@: set abbreviation and regular format	33
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	34	\@GLSuseri@: set regular format	38
\@Glsfirstplural@: bug fix: misspelt cs name	35	\@GLSuserii@: set regular format	38
\@Glsplural@: fixed bug \@Glsplural@ should be redefined not \@Glsplural	34	\@GLSuseriii@: set regular format	38
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	34	\@GLSuseriv@: set regular format	38
\glxtrtitlelongpl: bug fix: changed \glxtrlong to \glxtrlongpl ..	180	\@GLSuserv@: set regular format	39
\glxtrtitleshortpl: bug fix: changed \glxtrshort to \glxtrshortpl	175	\@GLSuservi@: set regular format	39
1.04 (2015-04-30)		\@gls@preglossaryhook: added check for entry's existence	118
short-em-footnote: renamed from "footnote-em"	167	\@glsdesc@: set abbreviation and regular format	36
1.04 (2016-05-02)		\@glsdescplural@: set abbreviation and regular format	36
\@@glxtrpostloctag: new	28	\@glsfirst@: set abbreviation and regular format	33
\@GLSdesc@: set abbreviation and regular format	36	\@glsfirstplural@: set abbreviation and regular format	34
\@GLSdescplural@: set abbreviation and regular format	36	\@glsname@: set abbreviation and regular format	35
\@GLSfirst@: set abbreviation format ..	33	\@glsplural@: set abbreviation and regular format	34
\@GLSfirstplural@: set abbreviation and regular format	35	\@glsymbol@: set regular format	37
\@GLSname@: set abbreviation and regular format	35		

\@glssymbolplural@: set regular format	37	short-em-nolong-desc: new	166
\@glstext@: set abbreviation and regular format	32	short-em-postfootnote: renamed from “postfootnote-em”	168
\@glstr@deprecated@abbrstyle: new	141	short-footnote: new	148
\@glstr@do@style: new	14	short-long-user: new	170
\@glstr@doloctag: new	28	short-long-user-desc: new	171
\@glstr@idx@entrynumberlist: switched from \let to \newcommand	80	short-nolong: new	151
\@glstr@pagetag: new	28	short-nolong-desc: new	152
\@glstr@pagetag: new	28	short-postfootnote: new	149
\@glstr@preloctag: new	28	short-sc-footnote: renamed from “footnote-sc”	158
\@glstr@preloctag: new	28	short-sc-nolong: new	156
\@glstr@preloctag: new	27, 28	short-sc-nolong-desc: new	157
\glossentrydesc: added glossdescfont attribute check	105	short-sc-postfootnote: renamed from “postfootnote-sc”	158
\Glossentryname: added glossnamefont attribute check	109	short-sm-footnote: renamed from “footnote-sm”	161
\glossentryname: added glossnamefont attribute check	107	short-sm-nolong: new	160
moved post name hook inside condition	109	short-sm-nolong-desc: new	160
\glsabbrvmfont: new	162	short-sm-postfootnote: renamed from “postfootnote-sm”	161
\glsabbrvuserfont: new	169	\letabbreviationstyle: new	141
\glsfirstabbrvmfont: new	162	\newabbreviationstyle: bug fix: corrected test for existence	140
\glsfirstabbrvuserfont: new	169	long-em-noshort-em: new	166
\glsfirstlongemfont: new	162	long-em-noshort-em-desc: new	167
\glsfirstlonguserfont: new	169	long-em-short-em: new	163
\glsifnotregularcategory: new	102	long-em-short-em-desc: new	163
\glslongdefaultfont: new	127	long-noshort: new	154
\glslongemfont: new	162	long-noshort-desc: new	154
\glslongfont: new	127	long-noshort-em: renamed from “long-em”	166
\glslonguserfont: new	169	long-noshort-em-desc: renamed from “long-desc-em”	167
\glstrassignfieldfont: new	32	long-noshort-sc: renamed from “long-sc”	157
\GlsXtrEnablePreLocationTag: new	27	long-noshort-sc-desc: renamed from “long-desc-sc”	157
\glstrfirstscfont: new	155	long-noshort-sm: renamed from “long-sm”	160
\glstrfirstsmfont: new	158	long-noshort-sm-desc: renamed from \long-desc-sm	161
\glstrlongshortdescsort: new	143	long-short-user: new	169
\glstrpostnamehook: added category check	111	long-short-user-desc: new	170
\glstrregularfont: new	29	\renewabbreviationstyle: new style: new	14
\glstruserfield: new	168		
\glstruserparen: new	168	1.05 (2016-06-10)	
\glstrusersuffix: new	169	\eglssetwidest: new	198
\GlsXtrWarnDeprecatedAbbrStyle: new	141	\glsFindWidestAnyName: new	200
short-em-long-em: new	164		
short-em-long-em-desc: new	165		
short-em-nolong: new	165		

\glsFindWidestAnyNameLocation: new	205	\@GLSfirst@: added check for nohyperfirst attribute	34
\glsFindWidestAnyNameSymbol: new	203	\@GLSfirstplural@: added check for nohyperfirst attribute	35
\glsFindWidestAnyNameSymbolLocation: new	204	\@GLSxtrp: new	54
\glsFindWidestLevelTwo: new	201	\@Glsfirst@: added check for nohyperfirst attribute	33
\glsFindWidestUsedAnyName: new	199	\@Glsfirstplural@: added check for nohyperfirst attribute	35
\glsFindWidestUsedAnyNameLocation: new	205	\@Glsxtrp: new	54
\glsFindWidestUsedAnyNameSymbol: new	202	\@gls@preglossaryhook: added \glossxtrsetpopts	118
\glsFindWidestUsedAnyNameSymbolLocation: new	203	\@glsfirst@: added check for nohyperfirst attribute	33
\glsFindWidestUsedLevelTwo: new	200	\@glsfirstplural@: added check for nohyperfirst attribute	34
\glsFindWidestUsedTopLevelName: new	199	\@glsxtrinmark: new	173
\glsfirstlongfootnotefont: new	146	\@glsxtrnotinmark: new	173
\glsgetwidestname: new	198	\@glsxtrp: new	54
\glsgetwidestsubname: new	199	\@glsxtrp@opt: new	53
\glslongfootnotefont: new	146	\glossxtrsetpopts: new	53
\glsxtrAltTreeIndent: new	198	\glsps: new	56
\glsxtralttreeInit: new	198	\glspt: new	56
\glsxtrAltTreePar: new	198	\glsxtr@entry@p: new	55
\glsxtrAltTreeSetHangIndent: new	206	\glsxtrabbrvfootnote: new	146
\glsxtrAltTreeSetSubHangIndent: new	206	\glsxtrchecknohyperfirst: new	33
\glsxtralttreeSubSymbolDescLocation: new	198	\glsxtrfieldtitlecasecs: new	105
\glsxtralttreeSymbolDescLocation: new	197	\glsxtrifinmark: new	173
\glsxtrComputeTreeIndent: new	206	\GLSxtrp: new	57
\glsxtrComputeTreeSubIndent: new	206	\Glsxtrp: new	56
\glsxtrtreeopindent: new	198	\glsxtrp: new	55
short-em-long: fixed incorrect font used by long form	164	\glsxtrsetpopts: new	53
\xglsssetwidest: new	198	short-long-desc: added text key	145
1.06 (2016-06-18)		fixed misspelling of \glsabbrvfont in plural key	145
\@glsdoifexist sorwarn: new	9	long-short-desc: added missing text key	143
\@glsxtr@docdefval: new	9	fixed misspelling of \glsabbrvfont	144
\@glsxtr@usesee: new	19	footnote: changed first forms to use \glsfirstlongfootnotefont	146
General: disabled docdef key at the start of the document	16	postfootnote: removed \footnote from first keys	148
docdef option changed to choice	9	switched from \glsfirstlongfont to \glsfirstlongfootnotefont	149
\glsxtr@usesee: new	19	\RestoreAcronyms: modified	
\glsxtrusesee: new	19	\@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse	74
\glsxtruseseeformat: new	19	1.08 (2016-12-13)	
\if@glsxtrdocdefrestricted: new	9	\@@glsxtr@record: new	6
1.07 (2016-08-15)			
\@@glsxtrp: new	53		

\@GLS@: added \@glxtr@record 31	1.09 (2016-12-16)	
\@GLSpl@: added \@glxtr@record	... 31		\@glxtr@gettype: new 78
\@Gls@: added \@glxtr@record 30		\@glxtr@mixed@assign@sortkey:
\@Glspl@: added \@glxtr@record	... 31		new 78
\@Gls@: added \@glxtr@record 30		\@printglossary: redefined to save
\@Gls@@link@: added			options 78
\@glxtr@record 31		\@glxtr@makeglossaries: new 78
\@Gls@field@link: added		1.10 (2016-12-17)	
\@glxtr@record 30		\@GLSpl@: fixed bug caused by typo in
\@Gls@saveentrycounter: new 15		command name 31
\@Glsdispl: added \@glxtr@record	. 31	1.11 (2017-01-19)	
\@Glspl@: added \@glxtr@record	... 30		\@glxtr@do@redef@forglsentries:
\@glxtr@dorecord: new 7		new 5
\@glxtr@err@undefaction: new 5		\@glxtr@noidx@do: new 87
\@glxtr@record: new 6		\@glxtr@redef@forglsentries: new . 5
\@glxtr@warn@onexistsordo: new	... 5		\@glxtr@shortcutsval: new 12
\@glxtr@warn@undefaction: new	... 5		\@glxtr@unsrt@getgrouptitle: new 87
\@print@unsrt@glossary: new 87		\@print@noidx@glossary: added
General: added record package option 8		redefinition 80
\@gladd: added \@glxtr@record 31		\@glxtr@addloclistfield: added
\@glsoifexists: now defines			group key 8
\@glslabel 17		added location key 7
\@glxtr@@do@wrglossary: new 15		\@glxtr@fields: new 86
\@glxtr@addloclistfield: new 7		\@glxtr@linkprefix: new 86
\@glxtr@indexonly@saveentrycounter:			\@glxtr@org@newignoredglossary:
new 7		new 17
\@glxtr@record: new 7		\@glxtr@s@newignoredglossary: new 17
\@glxtr@resource: new 86		\@glxtr@shortcutsval: new 86
\@glxtr@saveentrycounter: new 15		\@glxtr@texencoding: new 86
\@glxtr@setup@record: new 7		\@glxtr@writefields: new 86
\@glxtrassignfieldfont: added check			\@GlsXtrLoadResources: new 85
for existence 32		\@glxtrresourcefile: changed
\@glxtrresourcefile: new 85		extension to .glstex 85
\@printunsrtglossaries: new 86		\@newignoredglossary: added starred
\@printunsrtglossary: new 86		version 17

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	
\.	119, 197
\@cGLS@	61, 69
\@cGLSpl@	61, 69
\@cGLspl@	61, 69
\@cglS@	61, 69
\@cglSpl@	61, 63, 69
\@do@wrglossary	7, 76
\@do@wrglossary	8, 9, 15, 32
\@glo@assign@sortkey	78
\@glo@list	5
\@glo@type	87
\@glslocalreset	59
\@glslocalunset	59
\@glsreset	59
\@glsunset	58
\@glsxtr@autoindex@escspch	113–115
\@glsxtr@checkspch	112, 113, 115
\@glsxtr@disabledflycommand	25
\@glsxtr@record	8, 9
\@glsxtrp	54, 55
\@glsxtrpostloctag	27
\@glsxtrpreloctag	27, 28
\@newglossaryentry@defcounters	60
\@newglossaryentry@defunitcounters	68
\@par	198
\@ACRlong	50
\@ACRlongpl	50
\@ACRshort	50
\@ACRshortpl	50
\@Acrlong	50
\@Acrlongpl	50
\@Acrshort	50
\@Acrshortpl	50
\@GLS@	50, 63, 64
\@GLSdesc@	36
\@GLSpl@	50, 64, 65
\@GLSplural@	51
\@GLSsymbol@	37
\@GLStext@	51
\@GLSxtr@full	128
\@GLSxtr@fullpl	130
\@GLSxtr@p@acrlong@	50
\@GLSxtr@p@acrlongpl@	50
\@GLSxtr@p@acrshort@	50
\@GLSxtr@p@acrshortpl@	50
\@GLSxtr@p@long@	50
\@GLSxtr@p@longpl@	50
\@GLSxtr@p@plural@	50
\@GLSxtr@p@short@	50
\@GLSxtr@p@shortpl@	50
\@GLSxtr@p@text@	50
\@GLSxtrlong	50, 133
\@GLSxtrlongpl	50, 136
\@GLSxtrp	58
\@GLSxtrshort	50, 131
\@GLSxtrshortpl	50, 134
\@Gls@	50, 63, 64
\@Gls@acentryname	72
\@Gls@entry@field	44, 56, 57
\@Gls@entryname	72
\@GlsXtrEnableOnTheFly	22
\@Glspl@	50, 63, 64
\@Glsplural@	51
\@Glstext@	51
\@Glsxtr	23, 24
\@Glsxtr@full	128
\@Glsxtr@fullpl	129
\@Glsxtr@p@acrlong@	50
\@Glsxtr@p@acrlongpl@	50
\@Glsxtr@p@acrshort@	50
\@Glsxtr@p@acrshortpl@	50
\@Glsxtr@p@long@	50
\@Glsxtr@p@longpl@	50

\@Glsxtr@p@plural@	50	\@glo@parent	200–202
\@Glsxtr@p@short@	50	\@glo@see	19, 20
\@Glsxtr@p@shortpl@	50	\@glo@sort	112
\@Glsxtr@p@text@	50	\@glo@sorttype	77, 80
\@Glsxtrlong	50, 132	\@glo@thislettergrp	88
\@Glsxtrlongpl	50, 136	\@glo@thisvalue	168
\@Glsxtrp	57	\@glo@tmp	44
\@Glsxtrpl	24	\@glo@type	20, 72, 75, 78, 80, 81, 84, 85, 87
\@Glsxtrshort	50, 131	\@glo@types	102, 103, 199–205
\@Glsxtrshortpl	50, 134	\@glossary@default@style	25, 208
\@acrlong	50	\@gls@	50, 62, 64
\@acrlongpl	50	\@gls@@link	31
\@acrshort	50	\@gls@actualchar	112
\@acrshortpl	50	\@gls@adjustmode	31
\@alt@gls@hyp@opt	48	\@gls@alt@hyp@opt	48
\@auxout	7, 28, 61, 62, 70, 75, 81, 85, 86	\@gls@alt@hyp@opt@char	48
\@cGLS	64	\@gls@alt@hyp@opt@keys	48
\@cGLS@	61, 64, 69	\@gls@automake	77
\@cGLSpl	64	\@gls@checkedmkidx	112, 113, 115
\@cGLSpl@	61, 65, 69	\@gls@checkmkidxchars	112
\@cGlspl@	61, 69	\@gls@codepage	81
\@cgls@	61, 69	\@gls@counter	6, 7, 32
\@cglspl@	61, 69	\@gls@currentlettergroup	80, 87, 88
\@disable@onlypremakeg	75	\@gls@declareoption	4
\@do@auxoutstuff	81	\@gls@doautomake	77
\@do@newglossaryentry	72, 125	\@gls@encapchar	113
\@do@seeglossary	75	\@gls@entry@count	61, 62
\@empty	32, 40–43, 112, 113, 127–136	\@gls@entry@field	44, 55, 57, 58, 60
\@end@glxstr@addunused	20	\@gls@entry@unitcount	69, 70
\@end@glxstr@gettype	77, 78	\@gls@field@font	32–39
\@end@glxstr@usesee	19	\@gls@field@link	32–39, 44, 45
\@endfortrue	139	\@gls@getgrouptitle	87
\@firstofone	32, 106, 111, 117	\@gls@hyp@opt	44, 45, 48, 64, 127–136
\@firstofthree	32, 40–43, 48, 127, 129, 130, 132, 134, 135	\@gls@hyp@opt@cs	48
\@firstoftwo	33–37, 41, 43, 46, 48, 74, 120, 121, 128–130, 134–136, 173, 174	\@gls@increment@currcount	61
\@for	5, 14, 20, 60, 71, 75, 77, 87, 104, 116	\@gls@increment@currunitcount	69
\@glo@assign@sortkey	77	\@gls@keymap	7, 8, 19, 44, 86
\@glo@category	66	\@gls@label	7, 47, 48, 75, 76, 139
\@glo@counterprefix	7	\@gls@levelchar	113
\@glo@countunit	66	\@gls@link	30, 31, 40–44, 127–136
\@glo@default@sorttype	77	\@gls@link@checkfirsthyper	74
\@glo@group	8	\@gls@link@nocheckfirsthyper	30, 40–43, 127–136
\@glo@label	7, 8, 19, 20, 44, 199–206	\@gls@local@increment@currcount	61
\@glo@location	7, 8	\@gls@local@increment@currunitcount	69
\@glo@loclist	7	\@gls@location	87, 88
\@glo@lname	112	\@gls@loclist	78–80, 87, 88
\@glo@no@assign@sortkey	78	\@gls@longpl	122–124
		\@gls@noidx@do	80

<code>\@gls@noidx@nosanitizesort</code>	77	<code>\@glsxtr@autoindex@at</code>	112, 113
<code>\@gls@noidx@sanitizesort</code>	77	<code>\@glsxtr@autoindex@doextra@esc</code>	112
<code>\@gls@noidx@loclist@finalsep</code>	79	<code>\@glsxtr@autoindex@encap</code>	112–114
<code>\@gls@noidx@loclist@prev</code>	79	<code>\@glsxtr@autoindex@esc</code>	112, 114, 115
<code>\@gls@noidx@loclist@sep</code>	78, 79	<code>\@glsxtr@autoindex@escat</code>	112, 113
<code>\@gls@noref@warn</code>	76, 80	<code>\@glsxtr@autoindex@escencap</code> ...	113, 114
<code>\@gls@org@glsnoidx@displayloc</code>	79	<code>\@glsxtr@autoindex@esclevel</code> ...	113, 114
<code>\@gls@org@glssee@format</code>	79	<code>\@glsxtr@autoindex@escquote</code> ...	112, 114
<code>\@gls@preglossaryhook</code>	116	<code>\@glsxtr@autoindex@level</code>	113, 114
<code>\@gls@prevlevel</code>	207	<code>\@glsxtr@autoindex@setname</code>	112
<code>\@gls@quotechar</code>	112	<code>\@glsxtr@autoindex@crossrefs</code>	10, 19
<code>\@gls@reference</code>	21, 75	<code>\@glsxtr@cat</code>	60, 71, 116
<code>\@gls@saveentrycounter</code>	8, 9, 15, 32	<code>\@glsxtr@csname</code>	66, 67, 69
<code>\@gls@see@noindex</code>	85	<code>\@glsxtr@current@style</code>	25, 208
<code>\@gls@setdefault@glslink@opts</code>	47	<code>\@glsxtr@currentunitcount</code>	66, 67, 69
<code>\@gls@short</code>	124	<code>\@glsxtr@currunitcount</code>	68, 70
<code>\@gls@shortpl</code>	122, 124	<code>\@glsxtr@declareoption</code>	4, 10, 11, 13
<code>\@gls@sort</code>	88	<code>\@glsxtr@defaultnoglossarywarning</code> ..	13
<code>\@gls@tmpb</code>	115	<code>\@glsxtr@deprecated@abbrstyle</code>	157, 158, 161, 162, 166–168
<code>\@gls@type</code>	75–77, 139, 199–205	<code>\@glsxtr@disabledflycommand</code>	24
<code>\@gls@write@entrycounts</code>	61	<code>\@glsxtr@do@@@wrindex</code>	47, 48
<code>\@gls@write@entryunitcounts</code>	69	<code>\@glsxtr@do@glsdisablehyperinlist</code> ..	46
<code>\@gls@write@entryunitcounts@do</code>	70	<code>\@glsxtr@do@redef@for@gl@sentries</code>	6
<code>\@gls@abbrv@current@abbreviation</code> 123, 137		<code>\@glsxtr@do@style</code>	14, 190
<code>\@gls@acronymlists</code>	72	<code>\@glsxtr@do@titlecaps@warn</code> .	106–109, 117
<code>\@gls@disp</code>	31	<code>\@glsxtr@do@abbreviationsdef</code>	11
<code>\@gls@doifexists@orwarn</code>	9, 107–110	<code>\@glsxtr@do@accsupp</code>	13, 15
<code>\@gls@entry</code>	61, 62, 70	<code>\@glsxtr@docdefval</code>	9, 21
<code>\@gls@link</code>	49, 50	<code>\@glsxtr@doloctag</code>	27, 28
<code>\@gls@locref</code>	7	<code>\@glsxtr@dorecord</code>	7
<code>\@gls@numberformat</code>	6, 7, 32, 111, 112	<code>\@glsxtr@dostylewarn</code>	139
<code>\@gls@order</code>	75	<code>\@glsxtr@enabletagging</code>	116
<code>\@gls@pl@</code>	50, 63, 64	<code>\@glsxtr@end@</code>	22
<code>\@gls@plural@</code>	51	<code>\@glsxtr@endescpch</code>	112–115
<code>\@gls@punc@token</code>	121	<code>\@glsxtr@entrycount@org@localreset</code> .	61
<code>\@gls@style@alttree</code>	197	<code>\@glsxtr@entrycount@org@localunset</code> .	61
<code>\@gls@style@inline</code>	197	<code>\@glsxtr@entrycount@org@reset</code>	61
<code>\@gls@style@listdotted</code>	192	<code>\@glsxtr@entrycount@org@unset</code> ...	60, 61
<code>\@gls@target</code>	49	<code>\@glsxtr@entryunitcount@org@localreset</code>	69
<code>\@gls@text@</code>	51	<code>\@glsxtr@entryunitcount@org@localunset</code>	69
<code>\@gls@widestname</code>	198, 199, 206	<code>\@glsxtr@entryunitcount@org@reset</code> ..	69
<code>\@glsxtr</code>	22, 24	<code>\@glsxtr@entryunitcount@org@unset</code> 68, 69	
<code>\@glsxtr@@do@@@wrglossary</code>	76	<code>\@glsxtr@err@undefaction</code>	6, 8
<code>\@glsxtr@abbreviationsdef</code>	11, 15	<code>\@glsxtr@field@linkdefs</code>	30
<code>\@glsxtr@activate@initialtagging</code> ..	116, 118	<code>\@glsxtr@format@overridefalse</code>	111
<code>\@glsxtr@addunitcounter</code>	66	<code>\@glsxtr@format@overridetrue</code>	111
<code>\@glsxtr@addunusedxrefs</code>	20		
<code>\@glsxtr@attrval</code>	105–110, 112		

\@glxtr@foundinlist	121	\@glxtr@org@glxtrtitlefirst .	173, 174
\@glxtr@full	127	\@glxtr@org@glxtrtitlefirstplural	
\@glxtr@fullpl	129	173, 174
\@glxtr@gettype	77	\@glxtr@org@glxtrtitlefull ..	173, 175
\@glxtr@glossdescfont	105–107	\@glxtr@org@glxtrtitlefullpl	173, 175
\@glxtr@glossnamefont	107–110	\@glxtr@org@glxtrtitlelong ..	173, 174
\@glxtr@gobbleto@endescpch	115	\@glxtr@org@glxtrtitlelongpl	173, 174
\@glxtr@idx@displaynumberlist	76	\@glxtr@org@glxtrtitleplural	173, 174
\@glxtr@idx@entrynumberlist	76	\@glxtr@org@glxtrtitleshort .	173, 174
\@glxtr@ifcsstart	22	\@glxtr@org@glxtrtitleshortpl	173, 174
\@glxtr@ifpunctoken	121	\@glxtr@org@glxtrtitletext ..	173, 174
\@glxtr@ifunitcounter	66	\@glxtr@org@makeglossaries	74, 75
\@glxtr@insert@dots	123	\@glxtr@org@markboth	172, 173
\@glxtr@insert@dots@next	123	\@glxtr@org@markright	172, 173
\@glxtr@insertdots	124	\@glxtr@org@newacronymstyle	73, 74
\@glxtr@label	20, 104, 105	\@glxtr@org@postdescription	118
\@glxtr@loadstyles	191	\@glxtr@org@see@noindex	85
\@glxtr@mixed@assign@sortkey	77	\@glxtr@org@setacronymstyle	73, 74
\@glxtr@noidx@displaynumberlist ...	76	\@glxtr@org@printglossary	24, 78
\@glxtr@noidx@do	87	\@glxtr@org@warndep	122
\@glxtr@noidx@entrynumberlist	76	\@glxtr@p@acrlong@	50
\@glxtr@noidx@getgrouptitle	87	\@glxtr@p@acrlongpl@	50
\@glxtr@noidx@numberlistloop	77	\@glxtr@p@acrshort@	50
\@glxtr@notfoundinlist	121	\@glxtr@p@acrshortpl@	50
\@glxtr@optlist	24	\@glxtr@p@long@	50
\@glxtr@org@GLS@	31	\@glxtr@p@longpl@	50
\@glxtr@org@GLSpl@	31	\@glxtr@p@plural@	50
\@glxtr@org@GLs@	30	\@glxtr@p@short@	50
\@glxtr@org@GLspl@	31	\@glxtr@p@shortpl@	50
\@glxtr@org@Glxtrtitlefirst .	173, 174	\@glxtr@p@text@	50
\@glxtr@org@Glxtrtitlefirstplural		\@glxtr@pagetag	27, 28
.....	173, 174	\@glxtr@pagetag	27, 28
\@glxtr@org@Glxtrtitlefull ..	174, 175	\@glxtr@prevunitcount	68
\@glxtr@org@Glxtrtitlefullpl	174, 175	\@glxtr@printglossopts	24, 77, 78
\@glxtr@org@Glxtrtitlelong ..	173, 175	\@glxtr@record	8, 9, 30, 31
\@glxtr@org@Glxtrtitlelongpl	173, 175	\@glxtr@redef@forglsentries	6, 15
\@glxtr@org@Glxtrtitleplural	173, 174	\@glxtr@redefstyles	14, 190
\@glxtr@org@Glxtrtitleshort .	173, 174	\@glxtr@reg@glosslist	75–78
\@glxtr@org@Glxtrtitleshortpl	173, 174	\@glxtr@savepreloctag	27, 28
\@glxtr@org@Glxtrtitletext ..	173, 174	\@glxtr@setentrycountunsetattr .	59, 60
\@glxtr@org@MakeUppercase	173, 174	\@glxtr@setentryunitcountunsetattr	71
\@glxtr@org@checkfirsthyper	46, 74	\@glxtr@setupshortcuts	12, 13, 15
\@glxtr@org@delimN	28	\@glxtr@shortcutsval	12, 86
\@glxtr@org@delimR	28	\@glxtr@swaptwo	122
\@glxtr@org@doseeglossary	75	\@glxtr@tag	116
\@glxtr@org@glS@	30	\@glxtr@taggingcs	116
\@glxtr@org@glSdisp	31	\@glxtr@thisloctag	28
\@glxtr@org@glSignore	28	\@glxtr@tmp	14
\@glxtr@org@glSpl@	30	\@glxtr@type	105

<code>\Alp</code>	11	<code>\csdef</code>	44,
<code>\alp</code>	11		45, 61, 66, 67, 69, 100, 139–141, 148, 192
<code>\AnyTrackedLanguages</code>	189	<code>\csedef</code>	67
<code>\appto</code> 7, 8, 14, 19, 44, 47, 48, 60, 68, 111, 121, 123		<code>\csgdef</code>	17, 21, 27, 61, 67, 69, 70
<code>\AS</code>	11	<code>\csletcs</code>	141
<code>\As</code>	11	<code>\csname</code>	5, 6,
<code>\as</code>	11		25, 29, 32, 40–45, 54, 67, 75, 76, 81, 84,
<code>\ASP</code>	11		85, 87, 88, 105, 122, 127–137, 141, 142, 206
<code>\Asp</code>	11	<code>\csuse</code>	27, 44, 45, 55–57, 66–68, 70, 80,
<code>\asp</code>	11		87, 100, 111, 118, 119, 139–141, 199–202
<code>\AtBeginDocument</code>	16, 26	<code>\csxdef</code>	19, 67, 69
<code>\AtEndDocument</code>	19, 61, 69, 81	<code>\CurrentOption</code>	14, 191
B		<code>\CurrentTrackedTag</code>	189
<code>babel</code> package	112, 113, 120	<code>\CustomAbbreviationFields</code>	
<code>\begin</code>	80, 83, 87, 192–196		125, 142–146, 148,
<code>\begingroup</code>	6		150, 151, 153, 154, 163, 164, 166, 169, 170
C		D	
category attributes:		<code>\DeclareAcronymList</code>	72
<code>discardperiod</code>	120	<code>\DeclareOption</code>	4, 191
<code>entrycount</code>	58–61, 71	<code>\DeclareOptionX</code>	4, 14
<code>firstuc</code>	109	<code>\def</code>	6–8, 16, 19, 20, 22–24,
<code>glossdesc</code>	105		26, 29–43, 48, 50–53, 62–65, 72, 76–80,
<code>glossdescfont</code>	105		87, 111–117, 121–124, 127–136, 139, 207
<code>glossname</code>	107	<code>\defglentryfmt</code>	17
<code>glossnamefont</code>	107, 109	<code>\define@boolkey</code>	10, 46
<code>headuc</code>	175	<code>\define@choicekey</code>	6, 8, 9, 12, 13
<code>indexname</code>	112	<code>\define@key</code>	7, 8, 14, 44, 122
<code>indexonlyfirst</code>	47	<code>\DefineAcronymSynonyms</code>	12, 13
<code>insertdots</code>	124	<code>\delimN</code>	28
<code>nohyper</code>	46	<code>\delimR</code>	28
<code>nohyperfirst</code>	33–35	<code>\detokenize</code>	22
<code>regular</code>	29, 65, 142,	<code>\dimen@</code>	74, 199–206
	144–146, 148, 151–153, 163, 165, 169, 171	<code>\dimen@i</code>	200–202
<code>\cGLS</code>	11, 59, 71	<code>\dimen@ii</code>	200–202
<code>\cGls</code>	11, 59, 71	<code>\dimexpr</code>	26, 198
<code>\cglS</code>	11, 59, 71	<code>\disable@keys</code>	10, 16, 21
<code>\cGLSformat</code>	63	<code>\do</code>	5, 14, 20, 60, 71, 75, 77, 87, 104, 116
<code>\cGlsformat</code>	63	<code>\do@glS@link@checkfirsthyper</code>	
<code>\cglSformat</code>	62, 64		30, 31, 40–43, 127–136
<code>\cGLSpl</code>	11, 59, 71	<code>\do@glSdisablehyperinlist</code>	47
<code>\cGlspl</code>	11, 59, 71	<code>doc</code> package	114
<code>\cglSpl</code>	11, 59, 71	<code>\DTLifinlist</code>	75–78
<code>\cGLSplformat</code>	64	E	
<code>\cGlsplformat</code>	63	<code>\eappto</code>	14, 17, 112, 191
<code>\cglSplformat</code>	63, 65	<code>\edef</code> 5, 6, 17, 32, 46, 66, 67, 69, 75, 76, 78, 81,	
<code>\columnwidth</code>	26		105–108, 110, 112, 113, 115, 122, 200–202
<code>\count@</code>	61, 62, 70, 71	<code>\eglssetwidest</code>	199–206

<code>\glsabbrvemfont</code>	162–168	<code>\glsacspacemax</code>	74
<code>\glsabbrvfont</code>	51, 52, 73, 126, 130, 131, 134, 135, 137, 138, 142–151, 153–171	<code>\glsadd</code>	20, 82
<code>\glsabbrvuserfont</code>	169–171	<code>\glsaddstoragekey</code>	99
<code>\glsabbvfont</code> 142, 144, 146, 148, 163, 164, 169, 171	<code>\glsbackslash</code>	22
<code>\GLSaccessdesc</code>	36	<code>\glsapscase</code>	32–43, 45, 127–138
<code>\Glsaccessdesc</code>	36, 106, 115	<code>\glsacategory</code> 29, 32, 46, 51, 52, 100–102, 105– 111, 115, 116, 118, 119, 127–131, 133–135
<code>\glsaccessdesc</code>	36, 106, 119	<code>\glsacategorylabel</code>	46, 122–124, 148
<code>\GLSaccessdescplural</code>	36	<code>\glsaclosebrace</code>	83–85
<code>\Glsaccessdescplural</code>	36	<code>\glsacurrententrylabel</code> 26–28, 87, 111, 117, 118
<code>\glsaccessdescplural</code>	36	<code>\glsacurrentfieldvalue</code>	168
<code>\GLSaccessfirst</code>	34	<code>\glsacustomtext</code>	30, 40–43, 127–137, 139
<code>\Glsaccessfirst</code>	33	<code>\glsadefaulttype</code>	5, 10, 77, 82, 86
<code>\glsaccessfirst</code>	33	<code>\glsadescriptionaccessdisplay</code> ...	93, 106
<code>\GLSaccessfirstplural</code>	35	<code>\glsadescriptionpluralaccessdisplay</code> 93, 94
<code>\Glsaccessfirstplural</code>	35	<code>\glsadescwidth</code>	192, 194–196
<code>\glsaccessfirstplural</code>	34	<code>\glsadetoklabel</code> .	16–22, 32, 60, 61, 66–70, 75, 78–80, 87, 88, 105, 108, 109, 200–202
<code>\Glsaccesslong</code> 42, 125, 133, 143, 150, 153, 170		<code>\glsadisplaynumberlist</code>	76, 78
<code>\glsaccesslong</code>	42, 125, 132, 133, 143, 145, 147, 149, 150, 152–154, 170, 171	<code>\glsadohyperlink</code>	50
<code>\Glsaccesslongpl</code> 43, 126, 136, 143, 150, 153, 170	<code>\glsadoifexists</code> 9, 19, 30, 31, 40–43, 79, 80, 127–136
<code>\glsaccesslongpl</code>	43, 125, 135, 136, 143, 145, 147–150, 152–154, 170, 171	<code>\glsadoifexistsordo</code>	31
<code>\GLSaccessname</code>	35	<code>\glsadoifexistsorwarn</code> .	9, 105, 106, 115, 116
<code>\Glsaccessname</code>	35	<code>\glsadonohyperlink</code>	49, 50
<code>\glsaccessname</code>	35	<code>\glsadosanitizesort</code>	77
<code>\GLSaccessplural</code>	34	<code>\glsenableentrycount</code>	59, 62, 70
<code>\Glsaccessplural</code>	34	<code>\glsenableentryunitcount</code>	61, 71
<code>\glsaccessplural</code>	34	<code>\glsentrycurrcount</code>	60, 62, 68
<code>\Glsaccessshort</code> 40, 131, 137, 145, 147, 149, 152, 171	<code>\Glsentrydesc</code>	93, 98, 106
<code>\glsaccessshort</code>	40, 125, 130, 131, 137, 138, 143, 145, 147, 149–153, 170, 171	<code>\glsentrydesc</code>	93, 98, 107
<code>\Glsaccessshortpl</code> 41, 134, 137, 145, 147–149, 152, 171	<code>\Glsentrydescplural</code>	94, 98
<code>\glsaccessshortpl</code>	41, 125, 126, 134, 135, 137, 143, 145, 147, 149–153, 170, 171	<code>\glsentrydescplural</code>	94, 98
<code>\GLSaccesssymbol</code>	37	<code>\Glsentryfirst</code>	65, 91, 97
<code>\Glsaccesssymbol</code>	37, 116	<code>\glsentryfirst</code>	65, 91, 97, 186
<code>\glsaccesssymbol</code>	37, 115, 119	<code>\Glsentryfirstplural</code>	65, 91, 97
<code>\GLSaccesssymbolplural</code>	37	<code>\glsentryfirstplural</code> ...	65, 91, 92, 97, 186
<code>\Glsaccesssymbolplural</code>	37	<code>\glsentryfmt</code>	17
<code>\glsaccesssymbolplural</code>	37	<code>\Glsentryfull</code>	73
<code>\GLSaccessstext</code>	33	<code>\glsentryfull</code>	73
<code>\Glsaccessstext</code>	33	<code>\Glsentryfullpl</code>	73
<code>\glsaccessstext</code>	32	<code>\glsentryfullpl</code>	73
<code>\glsacrshortcutstrue</code>	12, 13	<code>\glsentryitem</code>	192–196, 207
		<code>\Glsentrylong</code>	52, 53, 65, 95, 99
		<code>\glsentrylong</code>	52, 53, 65, 95, 99, 148, 187
		<code>\Glsentrylongpl</code>	52, 53, 65, 96, 99

<code>\glsentrylongpl</code>	.. 52, 53, 65, 96, 99, 187, 188	<code>\glsfirstlonguserfont</code> 170, 171
<code>\Glsentryname</code> 89, 96, 107, 109, 110	<code>\GLSfirstplural</code> 179
<code>\glsentryname</code> 89, 96, 112, 199–206	<code>\Glsfirstplural</code> 179, 180
<code>\glsentrynumberlist</code> 76, 80, 204–206	<code>\glsfirstplural</code> 179
<code>\Glsentryplural</code> 90, 97	<code>\glsfirstpluralaccessdisplay</code> 91, 92
<code>\glsentryplural</code> 90, 97, 185	<code>\glsforeachincategory</code> 139
<code>\glsentryprevcount</code> 60, 62, 68	<code>\glsgenentryfmt</code> 29
<code>\glsentryprevmaxcount</code> 68	<code>\glsgetattribute</code>
<code>\glsentryprevtotalcount</code> 68 49, 62, 66–68, 105–108, 110, 112	
<code>\Glsentryshort</code> 51, 52, 94, 98	<code>\glsgetcategoryattribute</code> 100
<code>\glsentryshort</code>	... 51, 52, 74, 94, 98, 183, 184	<code>\glsgetwidestname</code> 198
<code>\Glsentryshortpl</code> 52, 53, 95, 99	<code>\glsgroupheading</code> 88, 192–196, 207
<code>\glsentryshortpl</code>	. 52, 53, 95, 98, 99, 183, 184	<code>\glsgroupskip</code> 88, 193–197, 208
<code>\Glsentrysymbol</code> 92, 97	<code>\glshasattribute</code>
<code>\glsentrysymbol</code> 92, 97, 203, 204 49, 61, 62, 66, 67, 69, 70, 105–	
<code>\Glsentrysymbolplural</code> 93, 98	111, 142, 144–146, 148, 163, 165, 169, 171	
<code>\glsentrysymbolplural</code> 92, 93, 98	<code>\glshascategoryattribute</code> 101
<code>\Glsentrytext</code> 90, 96	<code>\glshypernumber</code> 111
<code>\glsentrytext</code> 89, 90, 96, 97, 184, 185	<code>\glsifattribute</code> 33,
<code>\Glsentryuseri</code> 38	46, 47, 55, 103, 106–109, 117, 120, 175–183	
<code>\glsentryuseri</code> 38	<code>\glsifcategory</code> 103
<code>\Glsentryuserii</code> 38	<code>\glsifcategoryattribute</code>	. 46, 101, 102, 124
<code>\glsentryuserii</code> 38	<code>\glsifnotregular</code> 32
<code>\Glsentryuseriii</code> 38	<code>\glsifnotregularcategory</code> 102
<code>\glsentryuseriii</code> 38	<code>\glsifplural</code>	. 32, 34–37, 40–43, 120, 127–138
<code>\Glsentryuseriv</code> 39	<code>\glsifregular</code> 29, 32, 65
<code>\glsentryuseriv</code> 39	<code>\glsifregularcategory</code> 102
<code>\Glsentryuserv</code> 39	<code>\glsignore</code> 28
<code>\glsentryuserv</code> 39	<code>\glsinlinedescformat</code> 197
<code>\Glsentryuservi</code> 39	<code>\glsinlinesubdescformat</code> 197
<code>\glsentryuservi</code> 39	<code>\glsinsert</code> 32, 40–43, 127–138
<code>\glsfieldxdef</code> 104, 105	<code>\glskeylisttok</code> 72, 123, 125
<code>\glsfindwidesttoplevelname</code> 199	<code>\glslabel</code>	. 17, 29, 46, 49, 74, 119, 137, 138, 148
<code>\GLSfirst</code> 178, 179	<code>\glslabeltok</code> 72, 123, 125, 142–146,
<code>\Glsfirst</code> 179	148, 150, 151, 153, 154, 163–166, 169–171	
<code>\glsfirst</code> 178	<code>\glsletentryfield</code> 112
<code>\glsfirstabbrvdefaultfont</code>	<code>\glslink</code> 72, 73
..... 126, 143, 144, 147, 149–151, 153		<code>\glslink options</code>	
<code>\glsfirstabbrvmfont</code> 163–168	format 111
<code>\glsfirstabbrvfont</code>	hyper 172
..... 73, 125, 126, 142–153, 155–171		noindex 6, 46, 172
<code>\glsfirstabbrvuserfont</code> 170, 171	<code>\glslinkcheckfirsthyperhook</code> 46
<code>\glsfirstaccessdisplay</code> 91	<code>\glslinkvar</code> 48
<code>\glsfirstlongdefaultfont</code>	<code>\glslistdottedwidth</code> 192
..... 143, 144, 150, 152, 153		<code>\glslongaccessdisplay</code> 95
<code>\glsfirstlongemfont</code> 163–165, 167	<code>\glslongdefaultfont</code>
<code>\glsfirstlongfont</code>	... 125, 126, 142–145, 127, 143, 144, 146, 150, 152, 153	
147, 149, 150, 152–154, 163–167, 169–171		<code>\glslongemfont</code> 162–167
<code>\glsfirstlongfootnotefont</code> 146–149		

<code>\glslongfont</code>	52,	<code>\glsshortpluralaccessdisplay</code>	95
127, 132, 133, 135, 136, 143, 144, 147,		<code>\glsshorttok</code>	72, 123–125, 142–146,
149, 150, 152, 153, 163–165, 167, 170, 171		148, 150, 151, 154, 163, 164, 166, 169, 170	
<code>\glslongfootnotefont</code>	146, 147, 149	<code>\glssubentryitem</code>	192–197, 207
<code>\glslongpltok</code>	124, 125,	<code>\glssymbolaccessdisplay</code>	92
142–146, 153, 154, 163, 164, 166, 169, 171		<code>\glssymbolpluralaccessdisplay</code> ...	92, 93
<code>\glslongpluralaccessdisplay</code>	96	<code>\glstarget</code>	192–197, 207
<code>\glslongtok</code> ..	72, 123, 125, 142–146, 148,	<code>\GLstext</code>	177
150, 151, 153, 154, 163, 164, 166, 169, 170		<code>\Glstext</code>	177
<code>\glslonguserfont</code>	169–171	<code>\glstext</code>	177
<code>\glsnameaccessdisplay</code> ...	89, 107, 108, 110	<code>\glstextaccessdisplay</code>	89, 90
<code>\glsnamefont</code>	107, 109, 110	<code>\glstextformat</code>	31
<code>\glsnoidxdisplayloc</code>	79	<code>\glstextup</code>	155
<code>\glsnoidxdisplayloclisthandler</code>	79	<code>\glstreeindent</code>	206, 207
<code>\glsnoidxloclist</code>	80, 88	<code>\glstreenamebox</code>	207
<code>\glsnoidxnumberlistloophandler</code>	79	<code>\glstreenamefmt</code>	198–207
<code>\glsnonumberlistfalse</code>	26	<code>\glstype</code>	40–44, 127–136
<code>\glsnonumberlisttrue</code>	27	<code>\glunset</code>	20, 62–64
<code>\glsnumberlistloop</code>	76, 77	<code>\glswrite</code>	75
<code>\glsnumlistlastsep</code>	79	<code>\glswriteentry</code>	6
<code>\glsnumlistsep</code>	79	<code>\Glsxtr</code>	24
<code>\glsopenbrace</code>	83–85	<code>\glxtr</code>	24
<code>\glsorder</code>	75	<code>\glxtr@do@wrglossary</code>	7–9
<code>\glspagelistwidth</code>	192, 194–196	<code>\glxtr@addloclistfield</code>	8, 9
<code>\GLSpl</code>	59, 71	<code>\glxtr@addunused</code>	20
<code>\Glspl</code>	24, 59, 71	<code>\glxtr@applyabbrvfmt</code>	137
<code>\glspl</code>	23, 59, 71	<code>\glxtr@applyabbrvstyle</code> ...	122, 123, 139
<code>\GLSplural</code>	177, 178	<code>\glxtr@doption</code>	4, 10, 14, 15
<code>\Glsplural</code>	178	<code>\glxtr@fields</code>	86
<code>\glsplural</code>	177, 178	<code>\glxtr@headentry@p</code>	55, 56
<code>\glspluralaccessdisplay</code>	90	<code>\glxtr@ifnextpunc</code>	121
<code>\glspluralsuffix</code>	123, 127,	<code>\glxtr@ifpunctoken</code>	121
142, 144, 147, 149–151, 153, 155, 159, 169		<code>\glxtr@indexonly@saveentrycounter</code>	
<code>\glspostdescription</code>	118, 192–198	8, 9, 15
<code>\glspostinline</code>	197	<code>\glxtr@keylist</code>	22–24
<code>\glspostlinkhook</code> .	30, 31, 40–44, 54, 127–136	<code>\glxtr@linkprefix</code>	86
<code>\glsprestandardsort</code>	77	<code>\glxtr@makeglossaries</code>	75
<code>\glsresetentrylist</code>	80, 87	<code>\glxtr@next</code>	121
<code>\glsseeformat</code>	19, 75, 79	<code>\glxtr@org@newignoredglossary</code>	17
<code>\glssetabbrvfmt</code>	29, 32, 51,	<code>\glxtr@orgmakenoidxglossaries</code>	20
52, 105–110, 115, 116, 127–131, 133–135		<code>\glxtr@punclist</code>	121
<code>\glssetattribute</code> ...	142, 144–146, 148,	<code>\glxtr@record</code>	7
150, 151, 153, 154, 163, 165, 166, 169, 171		<code>\glxtr@resource</code>	85
<code>\glssetcategoryattribute</code>		<code>\glxtr@s@newignoredglossary</code>	17
.....	60, 71, 74, 100, 101, 103, 104, 116	<code>\glxtr@saveentrycounter</code>	6, 7
<code>\glssetnoexpandfield</code>	7, 8	<code>\glxtr@setup@record</code>	8, 15
<code>\glsshortaccessdisplay</code>	94	<code>\glxtr@shortcutsval</code>	86
<code>\glsshortpltok</code>	124, 125,	<code>\glxtr@texencoding</code>	86
142–146, 148, 150, 151, 163, 164, 169, 171		<code>\glxtr@usesee</code>	19

<code>\glxtr@warnonexistsordo</code>	6, 8, 9, 18	<code>\glxtrgenabbrvfmt</code>	29
<code>\glxtr@writefields</code>	85	<code>\Glsxtrheadfirst</code>	174
<code>\glxtrabbrvfootnote</code>	146–148	<code>\glxtrheadfirst</code>	174
<code>\glxtrabbrvtype</code>	10, 11, 125	<code>\Glsxtrheadfirstplural</code>	174
<code>\glxtraddallcrossrefs</code>	19	<code>\glxtrheadfirstplural</code>	174
<code>\glxtrAltTreeIndent</code>	198	<code>\Glsxtrheadfull</code>	174
<code>\glxtralttreeInit</code>	206	<code>\glxtrheadfull</code>	174
<code>\glxtrAltTreePar</code>	197	<code>\Glsxtrheadfullpl</code>	174
<code>\glxtrAltTreeSetHangIndent</code> ...	198, 207	<code>\glxtrheadfullpl</code>	174
<code>\glxtrAltTreeSetSubHangIndent</code> ...	207	<code>\Glsxtrheadlong</code>	174
<code>\glxtralttreeSubSymbolDescLocation</code>	207	<code>\glxtrheadlong</code>	174
<code>\glxtralttreeSymbolDescLocation</code> ..		<code>\Glsxtrheadlongpl</code>	174
.....	198, 207	<code>\glxtrheadlongpl</code>	174
<code>\glxtrassignfieldfont</code>	32–39	<code>\Glsxtrheadplural</code>	174
<code>\glxtrcat</code>	23, 24	<code>\glxtrheadplural</code>	174
<code>\glxtrchecknohyperfirst</code>	33–35	<code>\Glsxtrheadshort</code>	174
<code>\glxtrComputeTreeIndent</code>	207	<code>\glxtrheadshort</code>	174
<code>\glxtrComputeTreeSubIndent</code>	207	<code>\Glsxtrheadshortpl</code>	174
<code>\GlsXtrDefineAbbreviationShortcuts</code>		<code>\glxtrheadshortpl</code>	174
.....	12, 13	<code>\Glsxtrheadtext</code>	174
<code>\GlsXtrDefineOtherShortcuts</code>	12, 13	<code>\glxtrheadtext</code>	174
<code>\glxtrdiscardperiod</code>	119	<code>\glxtrtrifcounttrigger</code>	62–64
<code>\glxtrdoautoindexname</code>	47, 48, 111	<code>\glxtrtrifemptyglossary</code>	80, 84, 87
<code>\glxtrdopostpunc</code>	148	<code>\glxtrtrifindexing</code>	47
<code>\glxtrdowrglossaryhook</code>	47, 48	<code>\glxtrtrifinmark</code>	55–58, 173, 174
<code>\GlsXtrEnableEntryCounting</code>	71	<code>\glxtrtrifnextpunc</code>	121, 122
<code>\GlsXtrEnableEntryUnitCounting</code>	60	<code>\glxtrtrifperiod</code>	120
<code>\GlsXtrEnableOnTheFly</code>	22, 24, 25	<code>\glxtrtrifwasfirstuse</code>	32–35, 40–43, 46, 74, 119, 120, 128, 130–136, 148
<code>\glxtrfieldtitlecase</code>	106–109	<code>\Glsxtrinlinefullformat</code>	
<code>\glxtrfieldtitlecasecs</code>	105	126, 128, 140, 147, 149, 150, 152, 153, 188
<code>\glxtrfirstscfont</code>	155–158	<code>\glxtrinlinefullformat</code>	
<code>\glxtrfirstsmfont</code>	159–162	126–128, 140, 147, 149–153, 188
<code>\GlsXtrFormatLocationList</code>	27, 29, 204–206	<code>\Glsxtrinlinefullplformat</code>	
<code>\GLSxtrfull</code>	11, 181, 182	126, 129, 140, 147, 149, 150, 152, 153, 189
<code>\Glsxtrfull</code>	11, 182, 183	<code>\glxtrinlinefullplformat</code>	126,
<code>\glxtrfull</code>	11, 182	129, 130, 140, 147, 149, 150, 152, 153, 188
<code>\Glsxtrfullformat</code>	126, 138, 140, 143, 145, 147, 149, 151, 152, 154, 170, 171	<code>\glxtrininsertinsidefalse</code>	142
<code>\glxtrfullformat</code>	126, 138, 140, 143–145, 147, 149, 151–153, 170, 171	<code>\GLSxtrlong</code>	11, 180, 181
<code>\GLSxtrfullpl</code>	11, 182, 183	<code>\Glsxtrlong</code>	11, 181
<code>\Glsxtrfullpl</code>	11, 183	<code>\glxtrlong</code>	11, 180
<code>\glxtrfullpl</code>	11, 182	<code>\GLSxtrlongpl</code>	11, 180, 181
<code>\Glsxtrfullplformat</code>	126, 138, 140, 143, 145, 147, 149, 151, 152, 154, 170, 171	<code>\Glsxtrlongpl</code>	11, 181
<code>\glxtrfullplformat</code>	138, 140, 143, 145, 147, 149, 151, 152, 154, 170, 171	<code>\glxtrlongpl</code>	11, 180
<code>\glxtrfullsep</code>	125, 126, 142–145, 147–150, 152, 153, 163, 164, 168	<code>\glxtrlongshortdescsort</code>	143
		<code>\glxtrmarkhook</code>	172
		<code>\glxtrnewabbrevpresetkeyhook</code>	124
		<code>\glxtrnewnumber</code>	12
		<code>\glxtrnewsymbol</code>	12

<code>\glxstrNoGlossaryWarning</code>	13, 81	<code>\glxstrtagfont</code>	117
<code>\GlsXtrNoGlsWarningAutoMake</code>	84	<code>\Glsxtrtitlefirst</code>	173, 174, 186
<code>\GlsXtrNoGlsWarningBuildInfo</code>	85	<code>\glxstrtitlefirst</code>	173, 174, 186
<code>\GlsXtrNoGlsWarningCheckFile</code>	84	<code>\Glsxtrtitlefirstplural</code>	173, 174, 186, 187
<code>\GlsXtrNoGlsWarningEmptyMain</code>	84	<code>\glxstrtitlefirstplural</code>	173, 174, 186
<code>\GlsXtrNoGlsWarningEmptyNotMain</code>	84	<code>\Glsxtrtitlefull</code>	174, 175, 188
<code>\GlsXtrNoGlsWarningEmptyStart</code>	84	<code>\glxstrtitlefull</code>	173–175, 188
<code>\GlsXtrNoGlsWarningHead</code>	84	<code>\Glsxtrtitlefullpl</code>	174, 175, 189
<code>\GlsXtrNoGlsWarningMisMatch</code>	85	<code>\glxstrtitlefullpl</code>	173–175, 188, 189
<code>\GlsXtrNoGlsWarningNoOut</code>	85	<code>\Glsxtrtitlelong</code>	173–175, 187
<code>\GlsXtrNoGlsWarningTail</code>	85	<code>\glxstrtitlelong</code>	173, 174, 187
<code>\glxstrorg@ifKV@glslink@hyper</code>	30	<code>\Glsxtrtitlelongpl</code>	173–175, 188
<code>\GLSxtrp</code>	55	<code>\glxstrtitlelongpl</code>	173, 174, 187
<code>\Glsxtrp</code>	54	<code>\Glsxtrtitleplural</code>	173, 174, 185
<code>\glxstrp</code>	54, 56	<code>\glxstrtitleplural</code>	173, 174, 185
<code>\Glsxtrpl</code>	24	<code>\Glsxtrtitleshort</code>	173, 174, 184
<code>\glxstrpl</code>	24	<code>\glxstrtitleshort</code>	173, 174, 183
<code>\glxstrpostdescription</code>	103, 118, 197	<code>\Glsxtrtitleshortpl</code>	173, 174, 184
<code>\glxstrpostlink</code>	119	<code>\glxstrtitleshortpl</code>	173, 174, 183, 184
<code>\glxstrpostlinkendsentence</code>	119	<code>\Glsxtrtitletext</code>	173, 174, 185
<code>\glxstrpostlinkhook</code>	119	<code>\glxstrtitletext</code>	173, 174, 184, 185
<code>\glxstrpostlocalreset</code>	59, 61, 69	<code>\glxstrtreetopindent</code>	198, 206
<code>\glxstrpostlocalunset</code>	59, 61, 69	<code>\glxstrundefaction</code>	6, 8, 9, 16–18
<code>\glxstrpostnamehook</code>	108–110	<code>\glxstrundeftag</code>	16
<code>\GlsXtrPostNewAbbreviation</code>		<code>\GlsXtrUseAbbrStyleFmts</code>	
.....	125, 140, 142, 144–146, 148, 150, 151, 153, 154, 163, 165, 166, 169, 171	144, 146, 154–168, 170, 172
<code>\glxstrpostreset</code>	59, 61, 69	<code>\GlsXtrUseAbbrStyleSetup</code>	155–168, 170, 171
<code>\glxstrpostunset</code>	58, 60, 68, 69	<code>\glxstruserfield</code>	168
<code>\glxstrprotectlinks</code>	49, 50	<code>\glxstruserparen</code>	169–171
<code>\glxstrregularfont</code>	29, 32	<code>\glxstrusersuffix</code>	170, 171
<code>\glxstrresourcefile</code>	85	<code>\glxstruseseeformat</code>	19
<code>\glxstrrestoremarkhook</code>	172, 173	<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	122, 141
<code>\glxstrscfont</code>	155–158	<code>\GlsXtrWarning</code>	23, 24
<code>\glxstrscsuffix</code>	155–158		
<code>\GlsXtrSetActualChar</code>	114		
<code>\GlsXtrSetEncapChar</code>	115		
<code>\GlsXtrSetEscChar</code>	114		
<code>\GlsXtrSetLevelChar</code>	114		
<code>\glxstrsetpopts</code>	53		
<code>\glxstrsetupfulldefs</code>	127–130, 148		
<code>\GLSxtrshort</code>	11, 57, 58, 175, 176		
<code>\Glsxtrshort</code>	11, 176		
<code>\glxstrshort</code>	11, 175		
<code>\GLSxtrshortpl</code>	11, 175, 176		
<code>\Glsxtrshortpl</code>	11, 176		
<code>\glxstrshortpl</code>	11, 175, 176		
<code>\glxstrsmfont</code>	158–162		
<code>\glxstrsmsuffix</code>	159–162		

H

<code>\hangindent</code>	198, 206
<code>\hbox</code>	192
<code>\hfill</code>	192
<code>\href</code>	49
<code>\hsize</code>	25, 26
<code>\hss</code>	192
<code>\hyperlink</code>	49
<code>\hyperpage</code>	111
<code>\hyperref</code>	49
<code>hyperref package</code>	50, 111, 172, 183

I

<code>\if</code>	22
<code>\if@glxstr@format@override</code>	112
<code>\if@glxstrdocdefrestricted</code>	20

<code>\MakeTextUppercase</code>	174	<code>shortplural</code>	95, 99, 123
<code>\MakeUppercase</code>	173, 174	<code>symbol</code>	92, 97
<code>\markboth</code>	173	<code>symbolplural</code>	92, 93, 98
<code>\markright</code>	173	<code>text</code> 49, 89, 90, 96, 97, 142, 143, 176, 177, 184	
<code>\maxdimen</code>	26	<code>\newif</code>	111, 142
<code>\mbox</code>	207	<code>\newignoredglossary</code>	17
<code>\medskip</code>	84, 85	<code>\newlength</code>	198
<code>\MessageBreak</code>	21, 24, 62, 71, 77, 139	<code>\newnum</code>	12
<code>mfirstuc package</code>	116, 117	<code>\newrobustcmd</code>	44, 45, 53–55, 64, 116, 117, 127–136, 173, 175–183, 199–205
<code>\mfirstucMakeUppercase</code>	33– 43, 45, 51–53, 55, 57, 58, 64, 65, 73, 89– 99, 108, 109, 128, 130, 131, 133, 135–138	<code>\newsym</code>	12
<code>\mfu@checkword@arg</code>	117	<code>\newterm</code>	103
<code>\mfu@checkword@do</code>	117	<code>\newtoks</code>	123
N		<code>\newwrite</code>	75
<code>\NeedsTeXFormat</code>	4, 191	<code>\NoCaseChange</code>	55–58, 175–183
<code>\new@glossaryentry</code>	21, 77	<code>\noexpand</code>	14, 72, 81, 112, 113, 125, 191
<code>\new@ifnextchar</code> .	44, 45, 64, 65, 120, 127–136	<code>\nofiles</code>	84
<code>\newabbr</code>	11	<code>\noindent</code>	84, 85
<code>\newabbreviation</code>	11, 73	<code>\nopostdesc</code>	23, 24, 103
<code>\newabbreviationhook</code>	125	<code>\nr</code>	6, 8, 9, 12, 13
<code>\newabbreviationstyle</code>	142–146, 148, 150, 151, 153–171	<code>\ns@GLSxtrfull</code>	128
<code>\newacronym</code>	72, 73	<code>\ns@Glsxtrfull</code>	128
<code>\newacronymhook</code>	72	<code>\ns@glxtrfull</code>	127
<code>\newacronymstyle</code>	73, 74	<code>\ns@GLSxtrfullpl</code>	130
<code>\newcommand</code>	4–7, 9–20, 22–25, 27–29, 32, 33, 44, 45, 47, 48, 50, 53, 55–60, 62, 64–68, 70, 71, 73, 74, 78– 80, 82–87, 89–105, 111–123, 125–137, 139–141, 143, 146, 154, 155, 158, 159, 162, 168, 169, 173–189, 191, 197–199, 206	<code>\ns@Glsxtrfullpl</code>	129
<code>\newcount</code>	9	<code>\ns@glxtrfullpl</code>	129
<code>\newentry</code>	11	<code>\ns@GLSxtrlong</code>	133
<code>\newglossary</code>	10, 75	<code>\ns@Glsxtrlong</code>	132
<code>\newglossaryentry</code>	11, 21, 60, 68, 72, 103, 104, 125	<code>\ns@glxtrlong</code>	132
<code>\newglossaryentry options</code>		<code>\ns@GLSxtrlongpl</code>	136
<code>desc</code>	93, 98	<code>\ns@Glsxtrlongpl</code>	135, 136
<code>descplural</code>	93, 94, 98	<code>\ns@glxtrlongpl</code>	135
<code>first</code>	49, 91, 97, 142, 178–180, 185, 209	<code>\ns@GLSxtrshort</code>	131
<code>firstplural</code>	91, 97, 142, 179, 186, 209	<code>\ns@Glsxtrshort</code>	131
<code>long</code>	95, 99, 187	<code>\ns@glxtrshort</code>	130
<code>longplural</code>	96, 99, 187	<code>\ns@GLSxtrshortpl</code>	134
<code>name</code>	89, 96, 112	<code>\ns@Glsxtrshortpl</code>	134
<code>plural</code>	90, 97, 142, 177, 178, 185	<code>\ns@glxtrshortpl</code>	133
<code>see</code>	10, 19, 21, 75	<code>\null</code>	13
<code>short</code>	94, 98, 123	<code>\number</code>	67, 68, 70
		<code>\numexpr</code>	67, 70
O		O	
		<code>\or</code>	6, 8, 12, 13, 21
		<code>\org@glossaryentrynumbers</code>	26
P		P	
		<code>\p@gl@s@hyp@opt</code>	48
		<code>package options:</code>	
		<code>abbreviations</code>	10, 11

<code>\subglossentry</code>	88, 192–197, 207	<code>\unskip</code>	20, 192
		<code>\usepackage</code>	83–85
T			
<code>\tablehead</code>	195, 196	V	
<code>\tabletail</code>	195, 196	<code>\val</code>	6, 8, 9, 12, 13
<code>\tabularnewline</code>	192–197	W	
<code>\TeX</code>	82	<code>\warn@nomakeglossaries</code>	76
<code>\texorpdfstring</code>	55–57, 183–189	<code>\warn@noprintglossary</code>	76
<code>textcase</code> package	172	<code>\write</code>	61, 62, 70, 75, 81
<code>\textsc</code>	154	X	
<code>\textsmaller</code>	158	<code>\xcapitalisewords</code>	105
<code>\texttt</code>	82–85	<code>\xifinlist</code>	66
<code>\the</code>	72, 115, 125, 142–146, 148, 150, 151, 153, 154, 163–166, 169–171	<code>xindy</code>	209
<code>\theindex</code>	111	<code>xindy</code>	74
<code>\this@dialect</code>	189	<code>xkeyval</code> package	4
<code>\toks@</code>	115	<code>\XKV@checkchoice</code>	29
U		<code>\XKV@plfalse</code>	29
<code>\undef</code>	116	<code>\XKV@resa</code>	29
<code>\underline</code>	117	<code>\XKV@sttrue</code>	29