

glossaries-extra.sty v1.08: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2016-12-13

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	15
1.3	Modifications to Commands Provided by glossaries	15
1.3.1	Existence Checks	15
1.3.2	Document Definitions	18
1.3.3	Existing Glossary Style Modifications	23
1.3.4	Entry Formatting, Hyperlinks and Indexing	27
1.3.5	Entry Counting	56
1.3.6	Acronym Modifications	69
1.3.7	Indexing and Displaying Glossaries	72
1.4	Integration with glossaries-accsupp	82
1.5	Categories	92
1.6	Abbreviations	115
1.6.1	Abbreviation Styles Setup	132
1.6.2	Predefined Styles (Default Font)	135
1.6.3	Predefined Styles (Small Capitals)	147
1.6.4	Predefined Styles (Fake Small Capitals)	151
1.6.5	Predefined Styles (Emphasized)	155
1.6.6	Predefined Styles (User Parentheses Hook)	161
1.7	Using Entries in Headings	165
1.8	Multi-Lingual Support	182
2	Style Adjustments (glossaries-extra-stylemods.sty)	184
2.1	Package Initialisation	184
2.2	List-Like Styles	185
2.3	Longtable Styles	185
2.4	Long Ragged Styles	186
2.5	Supertabular Styles	188
2.6	Super Ragged Styles	189
2.7	Inline Style	190
2.8	Tree Styles	190
	Glossary	202
	Change History	203
	Index	212

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2016/12/13 v1.08 (NLCT)]
```

Requires xkeyval to define package options.

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

Requires etoolbox package.

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

Has glossaries already been loaded?

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

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

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

Not already loaded, so pass options to glossaries.

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

Set the defaults.

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

Declare package options.

`\glxtrundefaction` 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*{\glxtrundefaction}[2]{%
30   \@glxtrundeftag\PackageError{glossaries-extra}{#1}{#2}%
31 }

```

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

```

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

```

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

```

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

```

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 `\glxtrundefaction` should behave if `undefaction=warn` is set.

```

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

```

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

```

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

```

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

```

41 \newcommand*{\@glxtr@warn@onexistsordo}[1]{%
42   \GlossariesExtraWarning{glossaries-extra}{%
43     \string#1\space hasn't been defined, so
44     some errors won't be converted to warnings.
45     (This most likely means your version of
46     glossaries.sty is below version 4.19.))%
47 }

48 \define@choicekey{glossaries-extra.sty}{undefaction}[\val\nr]{%
49   {warn,error}%
50   {%
51     \ifcase\nr\relax
52       \let\glxtrundefaction\@glxtr@warn@undefaction
53       \let\glxtr@warnonexistsordo\@glxtr@warn@onexistsordo
54     \or
55       \let\glxtrundefaction\@glxtr@err@undefaction
56       \let\glxtr@warnonexistsordo\@gobble
57     \fi
58   }

```

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.

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

```
59 \newcommand*{\@glxtr@record}[2]{}
```

`@@glxtr@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.

```
60 \newcommand*{\@@glxtr@record}[2]{%
61   \begingroup
62   \def\@glsnumberformat{glsnumberformat}%
63   \ifcsdef{glo@#2@counter}%
64     {%
65       \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
66     }%
67     {%
```

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

```
68     \def\@gls@counter{page}%
69   }%
70   \setkeys{glslink}{#1}%
71   \ifKV@glslink@noindex
72     \else
73     \glswriteentry{#2}%
74     {%
```

Save the entry counter.

```
75     \glxtr@saveentrycounter
```

Temporarily redefine `\@@do@@wrglossary` so we can use `\glxtr@@do@wrglossary`.

```
76     \let\@@do@@wrglossary\@glxtr@dorecord
77     \glxtr@@do@wrglossary{#2}%
78   }%
79   \fi
80 \endgroup
81 }
```

`glxtr@dorecord`

```
82 \newcommand*\@glxtr@dorecord{%
83   \protected@write\@auxout{}\string\glxtr@record
84     {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
85     {\@glslocref}}%
86 }
```

`\glxtr@record`

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

tr@setup@record Initialise.

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

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

```
89 \newcommand*{\glxtr@indexonly@saveentrycounter}{%
90   \ifKV@glslink@noindex
91   \else
92     \glxtr@saveentrycounter
93   \fi
94 }
```

addloclistfield

```
95 \newcommand*{\glxtr@addloclistfield}{%
96   \key@ifundefined{glossentry}{loclist}{%
97     {%
98       \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
99       \appto\@gls@keymap{,{loclist}{loclist}}%
100      \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
101      \appto\@newglossaryentryposthook{%
102        \gls@assign@field{}{\@glo@label}{loclist}{\@glo@loclist}%
103      }%
104    }%
105  }%
106 }
```

Now define the record package option.

```
107 \define@choicekey{glossaries-extra.sty}{record}[\val\nr]%
108 {off,only,alsoindex}%
109 [only]%
110 {%
111   \ifcase\nr\relax
```

Don't record.

```
112   \def\glxtr@setup@record{%
113     \renewcommand*{\@glxtr@record}[2]{}%
114     \let\@do@wrglossary\glxtr@do@wrglossary
115     \let\@gls@saveentrycounter\glxtr@indexonly@saveentrycounter
116     \let\glxtrundefaction\glxtr@err@undefaction
117     \let\glxtr@warnonexistsordo\@gobble
118   }%
119   \or
```

Only record (don't index).

```
120   \def\glxtr@setup@record{%
121     \let\@glxtr@record\@glxtr@record
122     \let\@do@wrglossary\@gobble
123     \let\@gls@saveentrycounter\relax
124     \let\glxtrundefaction\glxtr@warn@undefaction
125     \let\glxtr@warnonexistsordo\glxtr@warn@onexistsordo
```

```

126      \glsxtr@addloclistfield
127    }%
128  \or

```

Record and index.

```

129    \def\glsxtr@setup@record{%
130      \let\@glsxtr@record\@glsxtr@record
131      \let\@do@wrglossary\glsxtr@do@wrglossary
132      \let\@gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
133      \let\glsxtrundefaction\@glsxtr@warn@undefaction
134      \let\glsxtr@warnonexistsordo\@glsxtr@warn@onexistsordo
135      \glsxtr@addloclistfield
136    }%
137  \fi
138 }

```

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.

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

```

139 \newcount\@glsxtr@docdefval

```

Need to provide conditional commands that are backward compatible:

if@glsxtrdocdef

```

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

```

lsxtrdocdeftrue

```

141 \newcommand*{\@glsxtrdocdeftrue}{\@glsxtr@docdefval=1 }

```

sxtrdocdeffalse

```

142 \newcommand*{\@glsxtrdocdeffalse}{\@glsxtr@docdefval=0 }

```

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

```

143 \define@choicekey{glossaries-extra.sty}{docdef}[\val\nr]%
144 {false,true,restricted}[true]%
145 {%
146   \@glsxtr@docdefval=\nr\relax
147   \ifnum\@glsxtr@docdefval=2\relax
148     \renewcommand*{\@glsdoifexistsorwarn}{\glsdoifexists}%
149   \fi
150 }

```

ocdefrestricted

```

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

```


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

```

152 \newcommand*{\@glsdoifexistsorwarn}{\glsdoifexistsorwarn}

```

indexcrossrefs Automatically index cross references at the end of the document

```

153 \define@boolkey{glossaries-extra.sty}{@glsxtr}{indexcrossrefs}[true]{%
154   \if@glsxtrindexcrossrefs
155   \else
156   \renewcommand*{\@glsxtr@autoindexcrossrefs}{}%
157   \fi
158 }

```

Switch off since this can increase the build time.

```

159 \@glsxtrindexcrossrefsfalse

```

But allow see key to switch it on automatically.

oindexcrossrefs

```

160 \newcommand*{\@glsxtr@autoindexcrossrefs}{\@glsxtrindexcrossrefstrue}

```

iesExtraWarning Allow users to suppress warnings.

```

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

```

raWarningNoLine Allow users to suppress warnings.

```

162 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
163   \PackageWarningNoLine{glossaries-extra}{#1}}

164 \@glsxtr@declareoption{nowarn}{%
165   \let\GlossariesExtraWarning\@gobble
166   \let\GlossariesExtraWarningNoLine\@gobble
167   \glsxtr@doption{nowarn}%
168 }

```

glsxtrabbrvtype Glossary type for abbreviations.

```

169 \newcommand*{\glsxtrabbrvtype}{\glsdefaulttype}

```

bbreviationsdef Set by abbreviations option.

```

170 \newcommand*{\@glsxtr@abbreviationsdef}{}

```

bbreviationsdef

```

171 \newcommand*{\@glsxtr@doabbreviationsdef}{%
172   \@ifpackageloaded{babel}%
173   {\providecommand{\abbreviationsname}{\acronymname}}%
174   {\providecommand{\abbreviationsname}{Abbreviations}}%
175   \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
176   \renewcommand*{\glsxtrabbrvtype}{abbreviations}%
177   \newcommand*{\printabbreviations}[1][1]{%
178     \printglossary[type=\glsxtrabbrvtype,##1]%

```

```

179 }%
180 \disable@keys{glossaries-extra.sty}{abbreviations}%
    If the acronym option hasn't been used, change \acronymtype to \glstrabbrvtype.
181 \ifglssacronym
182 \else
183 \renewcommand*{\acronymtype}{\glstrabbrvtype}%
184 \fi
185 }%

```

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

```

186 \@glstr@declareoption{abbreviations}{%
187 \let\@glstr@abbreviationsdef\@glstr@doabbreviationsdef
188 }

```

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

```

189 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
190 \newcommand*{\ab}{\cglS}%
191 \newcommand*{\abp}{\cglSpl}%
192 \newcommand*{\as}{\glstrshort}%
193 \newcommand*{\asp}{\glstrshortpl}%
194 \newcommand*{\al}{\glstrlong}%
195 \newcommand*{\alp}{\glstrlongpl}%
196 \newcommand*{\af}{\glstrfull}%
197 \newcommand*{\afp}{\glstrfullpl}%
198 \newcommand*{\Ab}{\cGlS}%
199 \newcommand*{\Abp}{\cGlSpl}%
200 \newcommand*{\As}{\GlsXtrshort}%
201 \newcommand*{\Asp}{\GlsXtrshortpl}%
202 \newcommand*{\Al}{\GlsXtrlong}%
203 \newcommand*{\Alp}{\GlsXtrlongpl}%
204 \newcommand*{\Af}{\GlsXtrfull}%
205 \newcommand*{\Afp}{\GlsXtrfullpl}%
206 \newcommand*{\AB}{\cGLS}%
207 \newcommand*{\ABP}{\cGLSpl}%
208 \newcommand*{\AS}{\GLSxtrshort}%
209 \newcommand*{\ASP}{\GLSxtrshortpl}%
210 \newcommand*{\AL}{\GLSxtrlong}%
211 \newcommand*{\ALP}{\GLSxtrlongpl}%
212 \newcommand*{\AF}{\GLSxtrfull}%
213 \newcommand*{\AFP}{\GLSxtrfullpl}%
214 \newcommand*{\newabbr}{\newabbreviation}%

```

Disable this command after it's been used.

```

215 \let\GlsXtrDefineAbbreviationShortcuts\relax
216 }

```

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

```

217 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
218   \newcommand*{\newentry}{\newglossaryentry}%
219   \ifdef\printsymbols
220   {%
221     \newcommand*{\newsym}{\glstrnewsymbol}%
222   }{}%
223   \ifdef\printnumbers
224   {%
225     \newcommand*{\newnum}{\glstrnewnumber}%
226   }{}%
227   \let\GlsXtrDefineOtherShortcuts\relax
228 }

```

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.

```

229 \newcommand*{\@glstr@setupshortcuts}{}

```

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.

```

230 \define@choicekey{glossaries-extra.sty}{shortcuts}[\val\nr]%
231 {acronyms,acro,abbreviations,abbr,other,all,true,false}[true]{%
232   \ifcase\nr\relax % acronyms
233     \renewcommand*{\@glstr@setupshortcuts}{%
234       \glstrshortcutstrue
235       \DefineAcronymSynonyms
236     }%
237   \or % acro
238     \renewcommand*{\@glstr@setupshortcuts}{%
239       \glstrshortcutstrue
240       \DefineAcronymSynonyms
241     }%
242   \or % abbreviations
243     \renewcommand*{\@glstr@setupshortcuts}{%
244       \GlsXtrDefineAbbreviationShortcuts
245     }%
246   \or % abbr
247     \renewcommand*{\@glstr@setupshortcuts}{%
248       \GlsXtrDefineAbbreviationShortcuts
249     }%
250   \or % other
251     \renewcommand*{\@glstr@setupshortcuts}{%
252       \GlsXtrDefineOtherShortcuts
253     }%
254   \or % all
255     \renewcommand*{\@glstr@setupshortcuts}{%

```

```

256     \glsacrshortcutstrue
257     \DefineAcronymSynonyms
258     \GlsXtrDefineAbbreviationShortcuts
259     \GlsXtrDefineOtherShortcuts
260   }%
261   \or % true
262     \renewcommand*{\@glxtr@setupshortcuts}{%
263       \glsacrshortcutstrue
264       \DefineAcronymSynonyms
265       \GlsXtrDefineAbbreviationShortcuts
266       \GlsXtrDefineOtherShortcuts
267     }%
268   \else % none, false
269     \renewcommand*{\@glxtr@setupshortcuts}{}%
270   \fi
271 }

```

glxtr@doaccsupp

```

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

```

accsupp If accsupp, load glossaries-accsupp package.

```

273 \@glxtr@declareoption{accsupp}{%
274   \renewcommand*{\@glxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}

```

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

```

275 \newcommand{\glxtrNoGlossaryWarning}[1]{%
276   \@glxtr@defaultnoglossarywarning{#1}%
277 }

```

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

```

278 \define@choicekey{glossaries-extra.sty}{nomissingglstext}[\val\nr]%
279 {true,false}[true]{%
280   \ifcase\nr\relax % true
281     \renewcommand{\glxtrNoGlossaryWarning}[1]{%
282       \null
283     }%
284   \else % false
285     \renewcommand{\glxtrNoGlossaryWarning}[1]{%
286       \@glxtr@defaultnoglossarywarning{#1}%
287     }%
288   \fi
289 }

```

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

glxtr@redefstyles

```

290 \newcommand*{\@glxtr@redefstyles}{}

```

stylemods

```

291 \define@key{glossaries-extra.sty}{stylemods}{%
292   \ifblank{#1}%
293   {%
294     \renewcommand*{\@glsxtr@redefstyles}{%
295       \RequirePackage{glossaries-extra-stylemods}}%
296   }%
297   {%
298     \renewcommand*{\@glsxtr@redefstyles}{%
299       \@for\@glsxtr@tmp:=#1\do{%
300         \IfFileExists{glossary-\@glsxtr@tmp.sty}%
301         {%
302           \eappto\@glsxtr@redefstyles{%
303             \noexpand\RequirePackage{glossary-\@glsxtr@tmp}}%
304         }%
305         {%
306           \PackageError{glossaries-extra}%
307             {Glossaries style package ‘glossary-\@glsxtr@tmp.sty’
308              doesn’t exist (did you mean to use the ‘style’ key?)}%
309             {The list of values (#1) in the ‘stylemods’ key should
310              match the glossary-xxx.sty files provided with
311              glossaries.sty}%
312         }%
313       }%
314     \appto\@glsxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
315   }%
316 }

```

glsxtr@do@style

```

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

```

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

```

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

```

Set this as the default style:

```

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

```

Set this style:

```

321   \setglossarystyle{#1}%
322 }%
323 }

```

Pass all other options to glossaries.

```

324 \DeclareOptionX*{%
325   \expandafter\glsxtr@doooption\expandafter{\CurrentOption}}

```

Process options.

```

326 \ProcessOptionsX

```

Load glossaries if not already loaded.

```
327 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
328 \@glsxtr@doaccsupp
```

Define abbreviations glossaries if required.

```
329 \@glsxtr@abbreviationsdef
330 \let\@glsxtr@abbreviationsdef\relax
```

Setup shortcuts if required.

```
331 \@glsxtr@setupshortcuts
```

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

```
332 \renewcommand{\glsxtr@doooption}[1]{\setupglossaries{#1}}%
```

Now define the user command:

```
333 \newcommand*{\glossariesextrasetup}[1]{%
334   \let\glsxtr@setup@record\relax
335   \let\@glsxtr@setupshortcuts\relax
336   \setkeys{glossaries-extra.sty}{#1}%
337   \@glsxtr@abbreviationsdef
338   \let\@glsxtr@abbreviationsdef\relax
339   \@glsxtr@setupshortcuts
340   \glsxtr@setup@record
341 }
```

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

```
342 \let\glsxtr@@do@wrglossary\@do@wrglossary
```

aveentrycounter Save original definition of \@gls@saveentrycounter.

```
343 \let\glsxtr@saveentrycounter\@gls@saveentrycounter
```

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

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

Set up record option if required.

```
345 \glsxtr@setup@record
```

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

```
346 \AtBeginDocument{%
347   \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
348   \def\@glsxtrundeftag{\glsxtrundeftag}%
349 }
```

1.2 Extra Utilities

`\ifemptyglossary` `\glstrifemptyglossary{<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.)

```
350 \newcommand{\glstrifemptyglossary}[3]{%
351   \ifglossaryexists{#1}%
352   {%
353     \ifcsstring{glolist@#1}{,}{#2}{#3}%
354   }%
355   {%
356     \glstrundefaction{Glossary type '#1' doesn't exist}{}%
357     #2%
358   }%
359 }
```

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.

1.3.1 Existence Checks

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

```
360 \renewcommand{\glstoifexists}[2]{%
361   \ifglstentryexists{#1}{#2}%
362   {%
```

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

```
363   \edef\glslabel{\glstetoklabel{#1}}%
364   \glstrundefaction{Glossary entry '\glslabel'
365     has not been defined}{You need to define a glossary entry before
366     you can reference it.}%
367   }%
368 }
```

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

```
369 \renewcommand{\glstoifnoexists}[2]{%
370   \ifglstentryexists{#1}{%
```

```

371 \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
372 has already been defined}{#2}%
373 }

```

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

```

374 \ifdef\glstoifexistsordo
375 {%
376 \renewcommand{\glstoifexistsordo}[3]{%
377 \ifglstryexists{#1}{#2}%
378 {%
379 \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
380 has not been defined}{You need to define a glossary entry
381 before you can use it.}%
382 #3%
383 }%
384 }%
385 }
386 {%
387 \glstr@warnonexistsordo\glstoifexistsordo
388 \newcommand{\glstoifexistsordo}[3]{%
389 \ifglstryexists{#1}{#2}%
390 {%
391 \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
392 has not been defined}{You need to define a glossary entry
393 before you can use it.}%
394 #3%
395 }%
396 }%
397 }

```

`\arynoexistsordo` Similarly for `\doifglossarynoexistsordo`.

```

398 \ifdef\doifglossarynoexistsordo
399 {%
400 \renewcommand{\doifglossarynoexistsordo}[3]{%
401 \ifglossaryexists{#1}%
402 {%
403 \glstrundefaction{Glossary type '#1' already exists}{#2}%
404 #3%
405 }%
406 {#2}%
407 }%
408 }
409 {%
410 \glstr@warnonexistsordo\doifglossarynoexistsordo
411 \newcommand{\doifglossarynoexistsordo}[3]{%
412 \ifglossaryexists{#1}%
413 {%
414 \glstrundefaction{Glossary type '#1' already exists}{#2}%

```



```

415      #3%
416    }%
417    {#2}%
418  }%
419 }
420

```

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

```

421 \appto\@newglossaryentryposthook{%
422   \ifdefvoid\@glo@see
423     {\csxdef{glo@\@glo@label @see}{}}%
424     {%
425       \csxdef{glo@\@glo@label @see}{\@glo@see}%
426       \@glxtr@autoindexcrossrefs
427     }%
428 }
429 \appto\@gls@keymap{,{see}{see}}

```

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

```

430 \newcommand*{\glxtrusesee}[1]{%
431   \glsdoifexists{#1}%
432   {%
433     \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
434     \ifdefempty\@glo@see
435       {}%
436       {%
437         \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
438       }%
439     }%
440 }

```

`\glxtr@usesee`

```

441 \newcommand*{\glxtr@usesee}[1][\seename]{%
442   \@glxtr@usesee[#1]%
443 }

```

`\@glxtr@usesee`

```

444 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
445   \glxtruseseeformat{#1}{#2}%
446 }

```

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

```

447 \newcommand*{\glxtruseseeformat}[2]{%
448   \glsseeformat{#1}{#2}%
449 }

```

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

```

450 \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.

```

451 \newcommand*\glstraddallcrossrefs{%
452   \forallglossaries{\@glo@type}%
453   {%
454     \forallsentries[\@glo@type]{\@glo@label}%
455     {%
456       \ifglused{\@glo@label}{\@glstraddunusedxrefs{\@glo@label}}{}%
457     }%
458   }%
459 }

```

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

```

460 \newcommand*\@glstraddunusedxrefs[1]{%
461   \letcs{\@glo@see}{glo\glstoklabel{#1}@see}%
462   \ifdefvoid\@glo@see
463   {%
464     {%
465       \expandafter\glstraddunused\@glo@see\@end\glstraddunused
466     }%
467   }

```

`\glstraddunused` Adds all the entries if they haven't been used.

```

468 \newcommand*\glstraddunused[1][]{%
469   \@glstraddunused
470 }

```

`\glstraddunused` Adds all the entries if they haven't been used.

```

471 \def\@glstraddunused#1\@end\glstraddunused{%
472   \@for\@glstr@label:=#1\do
473   {%
474     \ifglused{\@glstr@label}{}%
475     {%
476       \glstradd[format=glstrunusedformat]{\@glstr@label}%
477       \glstrunset{\@glstr@label}%
478       \@glstraddunusedxrefs{\@glstr@label}%
479     }%
480   }%
481 }

```

`\glstrunusedformat`

```

482 \newcommand*\glstrunusedformat[1]{\unskip}

```

1.3.2 Document Definitions

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

```

483 \let\glstr@orgmakenoidxglossaries\makenoidxglossaries

```

```

484 \renewcommand{\makenoidxglossaries}{%
485   \glstr@orgmakenoidxglossaries
486   \ifglstrdocdefrestricted

```

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

```

487   \renewcommand*{\@gls@reference}[3]{%
488     \ifcsundef{@glsref@##1}{\csgdef{@glsref@##1}{}}{}%
489     \ifinlistcs{##2}{@glsref@##1}%
490     {}%
491     {\listcsgadd{@glsref@##1}{##2}}%
492     \ifcsundef{glo@glstetoklabel{##2}@loclist}%
493     {\csgdef{glo@glstetoklabel{##2}@loclist}{}}%
494     {}%
495     \listcsgadd{glo@glstetoklabel{##2}@loclist}{##3}%
496   }%
497 \else

```

Disable document definitions.

```

498   \@glstrdocdeffalse
499 \fi
500 \disable@keys{glossaries-extra.sty}{docdef}%
501 }

```

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

```

502 \renewcommand*{\@gls@defdocnewglossaryentry}{%
503   \ifcase \@glstr@docdefval
504     docdef=false:
505     \renewcommand*{\newglossaryentry}[2]{%
506       \PackageError{glossaries-extra}{Glossary entries must
507       be \MessageBreak defined in the preamble with \MessageBreak
508       package option 'docdef=false'\MessageBreak(consider using
509       'docdef=restricted')}{Move your glossary definitions to
510       the preamble. You can also put them in a \MessageBreak separate file
511       and load them with \string\loadglsentries.}%
512     }%
513   \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.

```

513     \let\gls@checkseeallowed\relax
514     \let\newglossaryentry\new@glossaryentry
515   \or

```

Restricted mode just needs to allow the see value.

```

516     \let\gls@checkseeallowed\relax
517   \fi
518 }%

```

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`

```
519 \newcommand*{\GlsXtrEnableOnTheFly}{%
520   \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
521 }
```

`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.)

```
522 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
523   \renewcommand*{\glsdetoklabel}[1]{%
524     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
525     {%
526       \expandafter\detokenize\expandafter{##1}%
527     }%
528     {\detokenize{##1}}}%
529   }%
530   \@GlsXtrEnableOnTheFly
531 }
532 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
533   \expandafter\if\glsbackslash#1%
534     #3%
535   \else
536     #4%
537   \fi
538 }
```

`sxtrstarflywarn`

```
539 \newcommand*{\glsxtrstarflywarn}{%
540   \GlossariesExtraWarning{Experimental starred version of
541     \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
542     read the warnings in the glossaries-extra user manual)}}%
543 }
```

`rEnableOnTheFly`

```
544 \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`

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

```

\glsxtr
546 \newcommand*\glsxtr[1] [] {%
547 \def\glsxtr@keylist{##1}%
548 \@glsxtr
549 }

\@glsxtr
550 \newcommand*\@glsxtr[2] [] {%
551 \ifglentryexists{##2}%
552 {%
553 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
554 }%
555 {%
556 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
557 description={\nopostdesc},##1}%
558 }%
559 \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
560 }

\Glsxtr
561 \newcommand*\Glsxtr[1] [] {%
562 \def\glsxtr@keylist{##1}%
563 \@Glsxtr
564 }

\@Glsxtr
565 \newcommand*\@Glsxtr[2] [] {%
566 \ifglentryexists{##2}%
567 {%
568 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
569 }%
570 {%
571 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
572 description={\nopostdesc},##1}%
573 }%
574 \expandafter\Gls\expandafter[\glsxtr@keylist]{##2}%
575 }

\glsxtrpl
576 \newcommand*\glsxtrpl[1] [] {%
577 \def\glsxtr@keylist{##1}%
578 \@glsxtrpl
579 }

\@glsxtrpl
580 \newcommand*\@glsxtrpl[2] [] {%
581 \ifglentryexists{##2}%
582 {%

```

```

583     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
584 }%
585 {%
586     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
587         description={\nopostdesc},##1}%
588 }%
589 \expandafter\glspl\expandafter[\glsxtr@keylist]{##2}%
590 }

```

\Glsxtrpl

```

591 \newcommand*\Glsxtrpl[1][1]{%
592     \def\glsxtr@keylist{##1}%
593     \@Glsxtrpl
594 }

```

\@Glsxtrpl

```

595 \newcommand*\@Glsxtrpl[2][1]{%
596     \ifglstryexists{##2}
597     {%
598         \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
599     }%
600     {%
601         \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
602             description={\nopostdesc},##1}%
603     }%
604     \expandafter\Glspl\expandafter[\glsxtr@keylist]{##2}%
605 }

```

\GlsXtrWarning

```

606 \newcommand*\GlsXtrWarning[2]{%
607     \def\glsxtr@optlist{##1}%
608     \@onelevel@sanitize\glsxtr@optlist
609     \GlossariesExtraWarning{The options '\glsxtr@optlist' have
610         been ignored for entry '##2' as it has already been defined}%
611 }

```

Disable commands after the glossary:

```

612 \let\glsxtr@orgprintglossary\@printglossary
613 \renewcommand\@printglossary[2]{%
614     \glsxtr@orgprintglossary{##1}{##2}%
615     \def\glsxtr{\glsxtr@disabledflycommand\glsxtr}%
616     \def\glsxtrpl{\glsxtr@disabledflycommand\glsxtrpl}%
617     \def\Glsxtr{\glsxtr@disabledflycommand\Glsxtr}%
618     \def\Glsxtrpl{\glsxtr@disabledflycommand\Glsxtrpl}%
619 }

```

\glsxtr@disabledflycommand

```

620 \newcommand*\glsxtr@disabledflycommand[1]{%
621     \PackageError{glossaries-extra}%

```

```

622   {\string##1\space can't be used after any of the \MessageBreak
623     glossaries have been displayed}%
624   {The on-the-fly commands enabled by
625     \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
626     before the glossaries. If you want to use any entries \MessageBreak
627     after any of the glossaries, you must use the standard \MessageBreak
628     method of first defining the entry and then using the \MessageBreak
629     entry with commands like \string\gls}%
630     @@glxtr@disabledflycommand
631   }%
632   \newcommand*{@@glxtr@disabledflycommand}[2][\{##2}

```

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

```

633   \let\GlsXtrEnableOnTheFly\relax
634 }
635 \@onlypreamble\GlsXtrEnableOnTheFly

```

1.3.3 Existing Glossary Style Modifications

Modify \setglossarystyle to keep track of the current style. This allows the \glossaries-extra-stylemods package to reset the current style after the required modifications have been made.

r@current@style Initialise the current style to the default style.

```

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

```

Modify \setglossarystyle to set the above.

etglossarystyle

```

637 \renewcommand*{\setglossarystyle}[1]{%
638   \ifcsundef{@glsstyle@#1}%
639   {%
640     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
641   }%
642   {%
643     \csname @glsstyle@#1\endcsname

```

Only set the current style if it exists.

```

644     \protected@edef\@glxtr@current@style{#1}%
645   }%
646   \ifx\@glossary@default@style\relax
647     \protected@edef\@glossary@default@style{#1}%
648   \fi
649 }

```

In case we have an old version of glossaries:

```

650 \ifdef\@glossary@default@style
651 {}
652 {%
653   \let\@glossary@default@style\relax
654 }

```

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

```

655 \ifdef\glslistdottedwidth
656 {%
657   \ifdim\glslistdottedwidth=.5\hsize
658     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
659   \AtBeginDocument{%
660     \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
661       \setlength{\glslistdottedwidth}{.5\columnwidth}%
662     \fi
663   }%
664 \fi
665 }
666 {}%
```

Similarly for \glsdescwidth:

\glsdescwidth

```

667 \ifdef\glsdescwidth
668 {%
669   \ifdim\glsdescwidth=.6\hsize
670     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
671   \AtBeginDocument{%
672     \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
673       \setlength{\glsdescwidth}{.6\columnwidth}%
674     \fi
675   }%
676 \fi
677 }
678 {}%
```

and for \glspagelistwidth:

lspagelistwidth

```

679 \ifdef\glspagelistwidth
680 {%
681   \ifdim\glspagelistwidth=.1\hsize
682     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
683   \AtBeginDocument{%
684     \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
685       \setlength{\glspagelistwidth}{.1\columnwidth}%
686     \fi
687   }%
688 \fi
689 }
690 {}%
```

aryentrynumbers Has the nonumberlist option been used?

```

691 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
```



```

692 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
693   \glsnonnumberlistfalse
694   \renewcommand*{\glossaryentrynumbers}[1]{%
695     \ifglsentryexists{\glscurrententrylabel}%
696     {%
697       \@glsxtrpreloctag
698       \GlsXtrFormatLocationList{#1}%
699       \@glsxtrpostloctag
700       \gls@save@numberlist{#1}%
701     }{}%
702   }%
703 \else
704   \glsnonnumberlisttrue
705   \renewcommand*{\glossaryentrynumbers}[1]{%
706     \ifglsentryexists{\glscurrententrylabel}%
707     {%
708       \gls@save@numberlist{#1}%
709     }{}%
710   }%
711 \fi

```

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

```

712 \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`

```

713 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
714   \let\@glsxtrpreloctag\@glsxtrpreloctag
715   \let\@glsxtrpostloctag\@glsxtrpostloctag
716   \renewcommand*{\@glsxtr@pagetag}{#1}%
717   \renewcommand*{\@glsxtr@pagetag}{#2}%
718   \renewcommand*{\@glsxtr@savepreloctag}[2]{%
719     \csgdef{\@glsxtr@preloctag@##1}{##2}%
720   }%
721   \renewcommand*{\@glsxtr@doloctag}{%
722     \ifcsundef{\@glsxtr@preloctag@\glscurrententrylabel}%
723     {%
724       \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
725         Rerun required}%
726     }%
727     {%
728       \csuse{\@glsxtr@preloctag@\glscurrententrylabel}%
729     }%

```

```

730 }%
731 }
732 \@onlypreamble\GlsXtrEnablePreLocationTag

glxstrpreloctag
733 \newcommand*{\@glxstrpreloctag}{%
734   \let\@glxstr@org@delimN\delimN
735   \let\@glxstr@org@delimR\delimR
736   \let\@glxstr@org@glsgignore\glsgignore
   \gdef is required as the delimiters may occur inside a scope.
737   \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
738   \renewcommand*{\delimN}{%
739     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
740     \@glxstr@org@delimN}%
741   \renewcommand*{\delimR}{%
742     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
743     \@glxstr@org@delimR}%
744   \renewcommand*{\glsgignore}[1]{%
745     \gdef\@glxstr@thisloctag{\relax}%
746     \@glxstr@org@glsgignore{##1}}%
747   \@glxstr@doloctag
748 }

glxstrpreloctag
749 \newcommand*{\@glxstrpreloctag}{}

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

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

lsxtrpostloctag
752 \newcommand*{\@glxstrpostloctag}{%
753   \let\delimN\@glxstr@org@delimN
754   \let\delimR\@glxstr@org@delimR
755   \let\glsgignore\@glxstr@org@glsgignore
756   \protected@write\@auxout{%
757     {\string\@glxstr@savepreloctag{\glscurrententrylabel}\@glxstr@thisloctag}}%
758 }

lsxtrpostloctag
759 \newcommand*{\@glxstrpostloctag}{}

lsxtr@preloctag
760 \newcommand*{\@glxstr@savepreloctag}[2]{}
761 \protected@write\@auxout{%
762   \string\providecommand\string\@glxstr@savepreloctag[2]{}

```

glsxtr@doloctag

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

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

```
764 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
765   \XKV@plfalse
766   \XKV@sttrue
767   \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
768   {%
769     \csname glsnonumberlist\XKV@resa\endcsname
770     \ifglsnonumberlist
771       \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
772     \else
773       \def\glossaryentrynumbers##1{%
774         \@glsxtr@preloctag
775         \GlsXtrFormatLocationList{##1}%
776         \@glsxtr@postloctag
777         \gls@save@numberlist{##1}}%
778     \fi
779   }%
780 }
```

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.

```
781 \renewcommand*{\glsentryfmt}{%
782   \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}{}}%
783   \glsifregular{\glslabel}%
784   {\glsxtrregularfont{\glsgenentryfmt}}%
785   {%
786     \ifglshasshort{\glslabel}%
787     {\glsxtrgenabbrvfmt}%
788     {\glsxtrregularfont{\glsgenentryfmt}}%
789   }%
790 }
```

sxtrregularfont Font used for regular entries.

```
791 \newcommand*{\glsxtrregularfont}[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 `\glsxtrifwasfirstuse` etc to allow the postlink hook to work better. This now has an optional argument that sets up the defaults.

```
792 \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.

```
793 \@glsxtr@record{#2}{#3}%
794 \glsdoifexists{#3}%
795 {%
```

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).

```
796 \let\glsxtrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
797 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
798 \def\glscustomtext{#4}%
799 \@glsxtr@field@linkdefs
800 #1%
801 \@gls@link[#2]{#3}{#4}%
802 \let\ifKV@glslink@hyper\glsxtrorg@ifKV@glslink@hyper
803 }%
804 \glspostlinkhook
805 }
```

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

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

```
806 \let\@glsxtr@org@gls@\@gls@
807 \def\@gls@#1#2{%
808 \@glsxtr@record{#1}{#2}%
809 \@glsxtr@org@gls@{#1}{#2}%
810 }%
```

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

```
811 \let\@glsxtr@org@glspl@\@glspl@
812 \def\@glspl@#1#2{%
813 \@glsxtr@record{#1}{#2}%
814 \@glsxtr@org@glspl@{#1}{#2}%
815 }%
```

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

```
816 \let\@glsxtr@org@Gls@\@Gls@
817 \def\@Gls@#1#2{%
818 \@glsxtr@record{#1}{#2}%
819 \@glsxtr@org@Gls@{#1}{#2}%
820 }%
```

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

```
821 \let\@glxtr@org@Glspl@\@Glspl@
822 \def\@Glspl@#1#2{%
823   \@glxtr@record{#1}{#2}%
824   \@glxtr@org@Glspl@{#1}{#2}%
825 }%
```

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

```
826 \let\@glxtr@org@GLS@\@GLS@
827 \def\@GLS@#1#2{%
828   \@glxtr@record{#1}{#2}%
829   \@glxtr@org@GLS@{#1}{#2}%
830 }%
```

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

```
831 \let\@glxtr@org@GLSpl@\@GLSpl@
832 \def\@GLS@#1#2{%
833   \@glxtr@record{#1}{#2}%
834   \@glxtr@org@GLSpl@{#1}{#2}%
835 }%
```

`\@glsdisp1` Save the original definition and redefine.

```
836 \let\@glxtr@org@glsdisp\@glsdisp
837 \renewcommand*{\@glsdisp}[3][]{%
838   \@glxtr@record{#1}{#2}%
839   \@glxtr@org@glsdisp[#1]{#2}{#3}%
840 }
```

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

```
841 \renewcommand*{\@gls@link}[3][]{%
842   \@glxtr@record{#1}{#2}%
843   \glsdoifexistsordo{#2}%
844   {%
845     \let\do@gls@link@checkfirsthyper\relax
846     \@gls@link[#1]{#2}{#3}%
847   }%
848   {%
849     \glstextformat{#3}%
850   }%
851   \glspostlinkhook
852 }
```

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

```
853 \renewrobustcmd*{\glsadd}[2][]{%
854   \@gls@adjustmode
855   \@glxtr@record{#1}{#2}%
856   \glsdoifexists{#2}%
857   {%
```

```

858 \def\@glsnumberformat{glsnumberformat}%
859 \edef\@gls@counter{\csname glo@\glsdetoklabel{#2}@counter\endcsname}%
860 \setkeys{glossadd}{#1}%
861 \@gls@saveentrycounter
862 \@@do@wrglossary{#2}%
863 }%
864 }

```

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

```

865 \newcommand*{\@glsxtr@field@linkdefs}{%
866 \let\glsxtrifwasfirstuse\@secondoftwo
867 \let\glsifplural\@secondoftwo
868 \let\gls@scapscase\@firstofthree
869 \let\glsinsert\@empty
870 }

```

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

`assignfieldfont`

```

871 \newcommand*{\@glsxtrassignfieldfont}[1]{%
872 \ifglsentryexists{#1}%
873 {%
874 \ifgls@hasshort{#1}%
875 {%
876 \glssetabbrvfmt{\gls@category{#1}}%
877 \glsifregular{#1}%
878 {\let\@gls@field@font\glsxtrregularfont}%
879 {\let\@gls@field@font\@firstofone}%
880 }%
881 {%
882 \glsifnotregular{#1}%
883 {\let\@gls@field@font\@firstofone}%
884 {\let\@gls@field@font\glsxtrregularfont}%
885 }%
886 }%
887 {%
888 \let\@gls@field@font\@gobble
889 }%
890 }

```

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

```

891 \def\@glstext@#1#2[#3]{%
892 \glsxtrassignfieldfont{#2}%
893 \@gls@field@link{#1}{#2}{\@gls@field@font{\gls@accesstext{#2}#3}}%
894 }

```

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

```

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

```

```

896 \glstrassignfieldfont{#2}%
897 \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
898 {\@gls@field@font{\Glsaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
899 }

```

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

```

900 \def\@Glstext@#1#2[#3]{%
901 \glstrassignfieldfont{#2}%
902 \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
903 {\@gls@field@font{\Glsaccesstext{#2}#3}}%
904 }

```

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

`ecknohyperfirst`

```

905 \newcommand*\glstrchecknohyperfirst}[1]{%
906 \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
907 }

```

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

```

908 \def\@glsfirst@#1#2[#3]{%
909 \glstrassignfieldfont{#2}%
910 \gls@field@link
911 [\let\glstrifwasfirstuse\@firstoftwo
912 \glstrchecknohyperfirst{#2}%
913 ]{#1}{#2}%
914 {\@gls@field@font{\glsaccessfirst{#2}#3}}%
915 }

```

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

```

916 \def\@Glsfirst@#1#2[#3]{%
917 \glstrassignfieldfont{#2}%
918 \gls@field@link
919 [\let\glstrifwasfirstuse\@firstoftwo
920 \let\glscapscase\@secondofthree
921 \glstrchecknohyperfirst{#2}%
922 ]%
923 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
924 }

```

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

```

925 \def\@GLSfirst@#1#2[#3]{%
926 \glstrassignfieldfont{#2}%

```

Ensure that \GLSfirst honours the nohyperfirst attribute.

```
927 \@gls@field@link
928 [\let\glstrifwasfirstuse\@firstoftwo
929 \let\glscapscase\@thirdofthree
930 \glstrchecknohyperfirst{#2}%
931 ]%
932 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
933 }
```

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

```
934 \def\@glsplural@#1#2[#3]{%
935 \glstrassignfieldfont{#2}%
936 \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
937 {\@gls@field@font{\glsaccessplural{#2}#3}}%
938 }
```

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

```
939 \def\@Glsplural@#1#2[#3]{%
940 \glstrassignfieldfont{#2}%
941 \@gls@field@link
942 [\let\glsifplural\@firstoftwo
943 \let\glscapscase\@secondofthree
944 ]%
945 {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
946 }
```

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

```
947 \def\@GLSplural@#1#2[#3]{%
948 \glstrassignfieldfont{#2}%
949 \@gls@field@link
950 [\let\glsifplural\@firstoftwo
951 \let\glscapscase\@thirdofthree
952 ]%
953 {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
954 }
```

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

```
955 \def\glsfirstplural@#1#2[#3]{%
956 \glstrassignfieldfont{#2}%
957 \@gls@field@link
958 [\let\glstrifwasfirstuse\@firstoftwo
959 \let\glsifplural\@firstoftwo
960 \glstrchecknohyperfirst{#2}%
961 ]%
962 {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
963 }
```


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

```
964 \def\@Glsfirstplural@#1#2[#3]{%
965   \glstrassignfieldfont{#2}%
   Ensure that \glfirstplural honours the nohyperfirst attribute.
966   \@gls@field@link
967   [\let\glstrifwasfirstuse\@firstoftwo
968   \let\glstifplural\@firstoftwo
969   \let\glscapscase\@secondofthree
970   \glstrchecknohyperfirst{#2}%
971   ]%
972   {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
973 }
```

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

```
974 \def\@GLSfirstplural@#1#2[#3]{%
975   \glstrassignfieldfont{#2}%
   Ensure that \glfirstplural honours the nohyperfirst attribute.
976   \@gls@field@link
977   [\let\glstrifwasfirstuse\@firstoftwo
978   \let\glstifplural\@firstoftwo
979   \let\glscapscase\@thirdofthree
980   \glstrchecknohyperfirst{#2}%
981   ]%
982   {#1}{#2}%
983   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
984 }
```

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

```
985 \def\@glsname@#1#2[#3]{%
986   \glstrassignfieldfont{#2}%
987   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
988 }
```

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

```
989 \def\@Glsname@#1#2[#3]{%
990   \glstrassignfieldfont{#2}%
991   \@gls@field@link
992   [\let\glscapscase\@secondoftwo]{#1}{#2}%
993   {\@gls@field@font{\Glsaccessname{#2}#3}}%
994 }
```

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

```
995 \def\@GLSname@#1#2[#3]{%
996   \glstrassignfieldfont{#2}%
997   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
998   {#1}{#2}%
999   {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
1000 }
```

```

\@Glsdesc@
1001 \def\@Glsdesc@#1#2[#3]{%
1002   \glstrassignfieldfont{#2}%
1003   \@gls@field@link{#1}{#2}{\@gls@field@font{\@glsaccessdesc{#2}#3}}%
1004 }

\@GLSdesc@   First letter uppercase version.
1005 \def\@GLSdesc@#1#2[#3]{%
1006   \glstrassignfieldfont{#2}%
1007   \@gls@field@link
1008   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1009   {\@gls@field@font{\@GLSaccessdesc{#2}#3}}%
1010 }

\@GLSdesc@   All uppercase version.
1011 \def\@GLSdesc@#1#2[#3]{%
1012   \glstrassignfieldfont{#2}%
1013   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1014   {#1}{#2}{\@gls@field@font{\@GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
1015 }

@Glsdescplural@   No case-changing version.
1016 \def\@Glsdescplural@#1#2[#3]{%
1017   \glstrassignfieldfont{#2}%
1018   \@gls@field@link
1019   [\let\glscapscase\@secondoftwo
1020    \let\glsifplural\@firstoftwo
1021    ]{#1}{#2}{\@gls@field@font{\@glsaccessdescplural{#2}#3}}%
1022 }

@Glsdescplural@   First letter uppercase version.
1023 \def\@Glsdescplural@#1#2[#3]{%
1024   \glstrassignfieldfont{#2}%
1025   \@gls@field@link
1026   [\let\glscapscase\@secondoftwo
1027    \let\glsifplural\@firstoftwo
1028    ]{#1}{#2}{\@gls@field@font{\@Glsaccessdescplural{#2}#3}}%
1029 }

@GLSdescplural@   All uppercase version.
1030 \def\@GLSdesc@#1#2[#3]{%
1031   \glstrassignfieldfont{#2}%
1032   \@gls@field@link
1033   [\let\glscapscase\@thirdoftwo
1034    \let\glsifplural\@firstoftwo
1035    ]%
1036    {#1}{#2}%
1037    {\@gls@field@font{\@GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
1038 }

```

```

\@glssymbol@
1039 \def\@glssymbol@#1#2[#3]{%
1040   \glstrassignfieldfont{#2}%
1041   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
1042 }

\@Glsymbol@   First letter uppercase version.
1043 \def\@Glsymbol@#1#2[#3]{%
1044   \glstrassignfieldfont{#2}%
1045   \@gls@field@link
1046   [\let\glscapscase\@secondoftwo]%
1047   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
1048 }

\@GLSsymbol@   All uppercase version.
1049 \def\@GLSsymbol@#1#2[#3]{%
1050   \glstrassignfieldfont{#2}%
1051   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1052   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
1053 }

lssymbolplural@   No case-changing version.
1054 \def\@lssymbolplural@#1#2[#3]{%
1055   \glstrassignfieldfont{#2}%
1056   \@gls@field@link
1057   [\let\glscapscase\@secondoftwo
1058   \let\glsifplural\@firstoftwo
1059   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
1060 }

lssymbolplural@   First letter uppercase version.
1061 \def\@lssymbolplural@#1#2[#3]{%
1062   \glstrassignfieldfont{#2}%
1063   \@gls@field@link
1064   [\let\glscapscase\@secondoftwo
1065   \let\glsifplural\@firstoftwo
1066   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
1067 }

LSsymbolplural@   All uppercase version.
1068 \def\@LSsymbol@#1#2[#3]{%
1069   \glstrassignfieldfont{#2}%
1070   \@gls@field@link
1071   [\let\glscapscase\@thirdoftwo
1072   \let\glsifplural\@firstoftwo
1073   ]%
1074   {#1}{#2}%
1075   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
1076 }

```

\@Glsuseri@ First letter uppercase version.

```
1077 \def\@Glsuseri@#1#2[#3]{%
1078   \glstrassignfieldfont{#2}%
1079   \@gls@field@link
1080   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1081   {\@gls@field@font{\Glsentryuseri{#2}#3}}%
1082 }
```

\@GLSuseri@ All uppercase version.

```
1083 \def\@GLSuseri@#1#2[#3]{%
1084   \glstrassignfieldfont{#2}%
1085   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1086   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseri{#2}#3}}}%
1087 }
```

\@Glsuserii@ First letter uppercase version.

```
1088 \def\@Glsuserii@#1#2[#3]{%
1089   \glstrassignfieldfont{#2}%
1090   \@gls@field@link
1091   [\let\glscapscase\@secondoftwo]%
1092   {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
1093 }
```

\@GLSuserii@ All uppercase version.

```
1094 \def\@GLSuserii@#1#2[#3]{%
1095   \glstrassignfieldfont{#2}%
1096   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1097   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
1098 }
```

\@Glsuseriii@ First letter uppercase version.

```
1099 \def\@Glsuseriii@#1#2[#3]{%
1100   \glstrassignfieldfont{#2}%
1101   \@gls@field@link
1102   [\let\glscapscase\@secondoftwo]%
1103   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
1104 }
```

\@GLSuseriii@ All uppercase version.

```
1105 \def\@GLSuseriii@#1#2[#3]{%
1106   \glstrassignfieldfont{#2}%
1107   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1108   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
1109 }
```

\@Glsuseriv@ First letter uppercase version.

```
1110 \def\@Glsuseriv@#1#2[#3]{%
1111   \glstrassignfieldfont{#2}%
1112 }
```

```

1112 \@gls@field@link
1113 [\let\glscapscase\@secondoftwo]%
1114 {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
1115 }

```

\@GLSuseriv@ All uppercase version.

```

1116 \def\@GLSuseriv@#1#2[#3]{%
1117 \glstrassignfieldfont{#2}%
1118 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1119 {#1}{#2}%
1120 {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
1121 }

```

\@Glsuserv@ First letter uppercase version.

```

1122 \def\@Glsuserv@#1#2[#3]{%
1123 \glstrassignfieldfont{#2}%
1124 \@gls@field@link
1125 [\let\glscapscase\@secondoftwo]%
1126 {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
1127 }

```

\@GLSuserv@ All uppercase version.

```

1128 \def\@GLSuserv@#1#2[#3]{%
1129 \glstrassignfieldfont{#2}%
1130 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1131 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
1132 }

```

\@Glsuservi@ First letter uppercase version.

```

1133 \def\@Glsuservi@#1#2[#3]{%
1134 \glstrassignfieldfont{#2}%
1135 \@gls@field@link
1136 [\let\glscapscase\@secondoftwo]%
1137 {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}%
1138 }

```

\@GLSuservi@ All uppercase version.

```

1139 \def\@GLSuservi@#1#2[#3]{%
1140 \glstrassignfieldfont{#2}%
1141 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1142 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
1143 }

```

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

\@acrshort No case change.

```

1144 \def\@acrshort#1#2[#3]{%

```

```

1145 \glsdoifexists{#2}%
1146 {%
1147   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1148   \let\glxtrifwasfirstuse\@secondoftwo
1149   \let\gl@sifplural\@secondoftwo
1150   \let\glscapscase\@firstofthree
1151   \let\gl$insert\@empty
1152   \def\glscustomtext{%
1153     \acronymfont{\gl@saccesssshort{#2}}#3%
1154   }%
1155   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1156 }%
1157 \glspostlinkhook
1158 }

```

\@Acrshort First letter uppercase.

```

1159 \def\@Acrshort#1#2[#3]{%
1160   \glsdoifexists{#2}%
1161   {%
1162     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1163     \let\glxtrifwasfirstuse\@secondoftwo
1164     \let\gl@sifplural\@secondoftwo
1165     \let\glscapscase\@secondofthree
1166     \let\gl$insert\@empty
1167     \def\glscustomtext{%
1168       \acronymfont{\Glsaccesssshort{#2}}#3%
1169     }%
1170     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1171   }%
1172   \glspostlinkhook
1173 }

```

\@ACRshort All uppercase.

```

1174 \def\@ACRshort#1#2[#3]{%
1175   \glsdoifexists{#2}%
1176   {%
1177     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1178     \let\glxtrifwasfirstuse\@secondoftwo
1179     \let\gl@sifplural\@secondoftwo
1180     \let\glscapscase\@thirdofthree
1181     \let\gl$insert\@empty
1182     \def\glscustomtext{%
1183       \mfirstucMakeUppercase{\acronymfont{\gl@saccesssshort{#2}}#3}%
1184     }%
1185     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1186   }%
1187   \glspostlinkhook
1188 }

```

\@acrshortpl No case change.

```
1189 \def\@acrshortpl#1#2[#3]{%
1190   \glsdoifexists{#2}%
1191   {%
1192     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1193     \let\glxtrifwasfirstuse\@secondoftwo
1194     \let\gl@sifplural\@firstoftwo
1195     \let\glscapscase\@firstofthree
1196     \let\gl$insert\@empty
1197     \def\glscustomtext{%
1198       \acronymfont{\gl@saccesssshortpl{#2}}#3%
1199     }%
1200     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1201   }%
1202   \glspostlinkhook
1203 }
```

\@Acrshortpl First letter uppercase.

```
1204 \def\@Acrshortpl#1#2[#3]{%
1205   \glsdoifexists{#2}%
1206   {%
1207     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1208     \let\glxtrifwasfirstuse\@secondoftwo
1209     \let\gl@sifplural\@firstoftwo
1210     \let\glscapscase\@secondofthree
1211     \let\gl$insert\@empty
1212     \def\glscustomtext{%
1213       \acronymfont{\Glsaccesssshortpl{#2}}#3%
1214     }%
1215     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1216   }%
1217   \glspostlinkhook
1218 }
```

\@ACRshortpl All uppercase.

```
1219 \def\@ACRshortpl#1#2[#3]{%
1220   \glsdoifexists{#2}%
1221   {%
1222     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1223     \let\glxtrifwasfirstuse\@secondoftwo
1224     \let\gl@sifplural\@firstoftwo
1225     \let\glscapscase\@thirdofthree
1226     \let\gl$insert\@empty
1227     \def\glscustomtext{%
1228       \mfirstucMakeUppercase{\acronymfont{\gl@saccesssshortpl{#2}}#3}%
1229     }%
1230     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1231   }%
1232   \glspostlinkhook
```

1233 }

\@acrlong No case change.

```
1234 \def\@acrlong#1#2[#3]{%
1235   \glsdoifexists{#2}%
1236   {%
1237     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1238     \let\glxtrifwasfirstuse\@secondoftwo
1239     \let\glsifplural\@secondoftwo
1240     \let\glscapscase\@firstofthree
1241     \let\glsinsert\@empty
1242     \def\glscustomtext{%
1243       \acronymfont{\glsaccesslong{#2}}#3%
1244     }%
1245     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1246   }%
1247   \glspostlinkhook
1248 }
```

\@Acrlong First letter uppercase.

```
1249 \def\@Acrlong#1#2[#3]{%
1250   \glsdoifexists{#2}%
1251   {%
1252     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1253     \let\glxtrifwasfirstuse\@secondoftwo
1254     \let\glsifplural\@secondoftwo
1255     \let\glscapscase\@secondofthree
1256     \let\glsinsert\@empty
1257     \def\glscustomtext{%
1258       \acronymfont{\Glsaccesslong{#2}}#3%
1259     }%
1260     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1261   }%
1262   \glspostlinkhook
1263 }
```

\@ACRlong All uppercase.

```
1264 \def\@ACRlong#1#2[#3]{%
1265   \glsdoifexists{#2}%
1266   {%
1267     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1268     \let\glxtrifwasfirstuse\@secondoftwo
1269     \let\glsifplural\@secondoftwo
1270     \let\glscapscase\@thirdofthree
1271     \let\glsinsert\@empty
1272     \def\glscustomtext{%
1273       \mfirstucMakeUppercase{\acronymfont{\glsaccesslong{#2}}#3}%
1274     }%
1275     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1276   }%
1277 }
```



```

1276 }%
1277 \glspostlinkhook
1278 }

```

\@acrlongpl No case change.

```

1279 \def\@acrlongpl#1#2[#3]{%
1280 \glsdoidexists{#2}%
1281 {%
1282 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1283 \let\glxtrifwasfirstuse\@secondoftwo
1284 \let\gl@sifplural\@firstoftwo
1285 \let\glscapscase\@firstofthree
1286 \let\gl$insert\@empty
1287 \def\glscustomtext{%
1288 \acronymfont{\gl@saccesslongpl{#2}}#3%
1289 }%
1290 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1291 }%
1292 \glspostlinkhook
1293 }

```

\@Acrlongpl First letter uppercase.

```

1294 \def\@Acrlongpl#1#2[#3]{%
1295 \glsdoidexists{#2}%
1296 {%
1297 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1298 \let\glxtrifwasfirstuse\@secondoftwo
1299 \let\gl@sifplural\@firstoftwo
1300 \let\glscapscase\@secondofthree
1301 \let\gl$insert\@empty
1302 \def\glscustomtext{%
1303 \acronymfont{\Glsaccesslongpl{#2}}#3%
1304 }%
1305 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1306 }%
1307 \glspostlinkhook
1308 }

```

\@ACRlongpl All uppercase.

```

1309 \def\@ACRlongpl#1#2[#3]{%
1310 \glsdoidexists{#2}%
1311 {%
1312 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1313 \let\glxtrifwasfirstuse\@secondoftwo
1314 \let\gl@sifplural\@firstoftwo
1315 \let\glscapscase\@thirdofthree
1316 \let\gl$insert\@empty
1317 \def\glscustomtext{%
1318 \mfirstucMakeUppercase{\acronymfont{\gl@saccesslongpl{#2}}#3}%

```

```

1319 }%
1320 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1321 }%
1322 \glspostlinkhook
1323 }

```

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

\@glsaddkey

```

1324 \renewcommand*{\@glsaddkey}[7]{%
1325   \key@ifundefined{glossentry}{#1}%
1326   {%
1327     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1328     \appto\@gls@keymap{,{#1}{#1}}%
1329     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
1330     \appto\@newglossaryentryposthook{%
1331       \letcs{\@glo@tmp}{@glo@#1}%
1332       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1333     }%
1334     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1335     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

1336   \ifcsdef{@gls@user@#1@}%
1337   {%
1338     \PackageError{glossaries}%
1339     {Can't define '\string#5' as helper command
1340     '\expandafter\string\csname @gls@user@#1@\endcsname' already
1341     exists}%
1342   }%
1343 }%
1344 {%
1345   \expandafter\newcommand\expandafter*\expandafter
1346   {\csname @gls@user@#1\endcsname}[2][ ]{%
1347     \new@ifnextchar[%
1348       {\csuse{@gls@user@#1@}{##1}{##2}}%
1349       {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
1350   \csdef{@gls@user@#1@}##1##2[##3]{%
1351     \@gls@field@link{##1}{##2}{#3{##2}##3}%
1352   }%
1353   \newrobustcmd*{#5}{%
1354     \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
1355 }%

```

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

```

1356   \ifcsdef{@Gls@user@#1@}%
1357   {%
1358     \PackageError{glossaries}%
1359     {Can't define '\string#6' as helper command

```

```

1360      '\expandafter\string\csname @Gls@user@#1@\endcsname' already
1361      exists}%
1362  {}%
1363 }%
1364 {%
1365   \expandafter\newcommand\expandafter*\expandafter
1366   {\csname @Gls@user@#1@\endcsname}[2] []{%
1367     \new@ifnextchar[%
1368       {\csuse{@Gls@user@#1@}{##1}{##2}}%
1369       {\csuse{@Gls@user@#1@}{##1}{##2} []}}%
1370   \csdef{@Gls@user@#1@}##1##2[##3]{%
1371     \@gls@field@link[\let\gls@caps@case\@secondofthree]%
1372     {##1}{##2}{#4{##2}##3}%
1373   }%
1374   \newrobustcmd*{#6}{%
1375     \expandafter\@gls@hyp@opt\csname @Gls@user@#1@\endcsname}%
1376   }%

```

Finally the all caps version (modified):

```

1377   \ifcsdef{@GLS@user@#1@}%
1378   {%
1379     \PackageError{glossaries}%
1380     {Can't define '\string#7' as helper command
1381     '\expandafter\string\csname @GLS@user@#1@\endcsname' already
1382     exists}%
1383   {}%
1384 }%
1385 {%
1386   \expandafter\newcommand\expandafter*\expandafter
1387   {\csname @GLS@user@#1@\endcsname}[2] []{%
1388     \new@ifnextchar[%
1389       {\csuse{@GLS@user@#1@}{##1}{##2}}%
1390       {\csuse{@GLS@user@#1@}{##1}{##2} []}}%
1391   \csdef{@GLS@user@#1@}##1##2[##3]{%
1392     \@gls@field@link[\let\gls@caps@case\@thirdofthree]%
1393     {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1394   }%
1395   \newrobustcmd*{#7}{%
1396     \expandafter\@gls@hyp@opt\csname @GLS@user@#1@\endcsname}%
1397   }%
1398 }%
1399 {%
1400   \PackageError{glossaries-extra}{Key '#1' already exists}{}%
1401 }%
1402 }

```

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

```

1403 \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.

```
1404 \let\@glstr@org@checkfirsthyper\@gls@link@checkfirsthyper
1405 \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.

```
1406 \ifglsused{\glslabel}%
1407 {\let\glstrifwasfirstuse\@secondoftwo}
1408 {\let\glstrifwasfirstuse\@firstoftwo}%
```

Store the category label for convenience.

```
1409 \edef\glscategorylabel{\glscategory{\glslabel}}%
1410 \ifglsused{\glslabel}%
1411 {%
1412   \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
1413   {\KV@glslink@hyperfalse}{}%
1414 }%
1415 {%
1416   \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
1417   {\KV@glslink@hyperfalse}{}%
1418 }%
1419 \glslinkcheckfirsthyperhook
1420 }
```

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.

```
1421 \ifdef\do@glstdisablehyperinlist
1422 {%
1423   \let\@glstr@do@glstdisablehyperinlist\do@glstdisablehyperinlist
1424   \renewcommand*{\do@glstdisablehyperinlist}{%
1425     \@glstr@do@glstdisablehyperinlist
1426     \glsifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}%
1427   }
1428 }
1429 {}
```

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

```
1430 \define@boolkey{glslink}{noindex}[true]{}
1431 \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

```
1432 \ifdef\@gls@setdefault@glslink@opts
1433 {
1434   \renewcommand*{\@gls@setdefault@glslink@opts}{%
1435     \KV@glslink@noindexfalse
```

```

1436 }
1437 }
1438 {
    Not defined so prepend it to \do@glssdisablehyperinlist to achieve the same effect.
1439 \newcommand*{\@gls@setdefault@glslink@opts}{%
1440 \KV@glslink@noindexfalse
1441 }
1442 \preto\do@glssdisablehyperinlist{\@gls@setdefault@glslink@opts}
1443 }

```

DefaultGlsOpts Set the default options for \glslink etc.

```

1444 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
1445 \renewcommand*{\@gls@setdefault@glslink@opts}{\setkeys{glslink}{#1}}%
1446 }

```

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

```

1447 \newcommand*{\glstrifindexing}[2]{%
1448 \ifKV@glslink@noindex #2\else #1\fi
1449 }

```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```

1450 \renewcommand*{\glswriteentry}[2]{%
1451 \glstrifindexing
1452 {%
1453 \ifglssindexonlyfirst
1454 \ifglssused{#1}
1455 {\glstrdoautoindexname{#1}{dualindex}}%
1456 {#2}%
1457 \else
1458 \glssifattribute{#1}{indexonlyfirst}{true}%
1459 {\ifglssused{#1}
1460 {\glstrdoautoindexname{#1}{dualindex}}%
1461 {#2}}%
1462 {#2}%
1463 \fi
1464 }%
1465 {}%
1466 }

```

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

```

1467 \appto\@do@@wrglossary{\@glstrdo@@wrindex
1468 \glstrdowrglossaryhook{\@gls@label}%
1469 }

```

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

Similarly for the “noidx” version:

s@noidxglossary

```
1470 \appto\gls@noidxglossary{\@glstr@do@@wrindex
1471 \glstrdowrglossaryhook{\@gls@label}}%
1472 }
```

xtr@do@@wrindex

```
1473 \newcommand*{\@glstr@do@@wrindex}{%
1474 \glstrdoautoindexname{\@gls@label}{dualindex}}%
1475 }
```

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.)

```
1476 \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.

```
1477 \newcommand*{\@gls@alt@hyp@opt}[1]{%
1478 \let\glslinkvar\@firstofthree
1479 \let\@gls@hyp@opt@cs#1\relax
1480 \@ifstar{\s@gls@hyp@opt}%
1481 {\@ifnextchar+%
1482 {\@firstoftwo{\p@gls@hyp@opt}}%
1483 {%
1484 \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
1485 {\@firstoftwo{\@alt@gls@hyp@opt}}%
1486 {\#1}%
1487 }%
1488 }%
1489 }
```

alt@gls@hyp@opt User version

```
1490 \newcommand*{\@alt@gls@hyp@opt}[1][ ]{%
1491 \let\glslinkvar\@firstofthree
1492 \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}
```

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

```
1493 \newcommand*{\@gls@alt@hyp@opt@char}{}
```

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

```
1494 \newcommand*{\@gls@alt@hyp@opt@keys}{}
```

rSetAltModifier

```
1495 \newcommand*{\GlsXtrSetAltModifier}[2]{%
1496 \let\@gls@hyp@opt\@gls@alt@hyp@opt
1497 \def\@gls@alt@hyp@opt@char{\#1}%
1498 \def\@gls@alt@hyp@opt@keys{\#2}%
1499 }
```

`\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 `\glentryshort` and, similarly, the long form commands like `\acrlong` and `\glxtrlong` to use `\glentrylong`.

```
1500 \renewcommand*{\glsdohyperlink}[2]{%
1501   \hyperlink{#1}{\glxtrprotectlinks#2}}
```

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

```
1502 \ifundef\glsdonohyperlink
1503 {%
1504   \renewcommand{\glsdisablehyper}{%
1505     \KV@glslink@hyperfalse
1506     \let\@glslink\glsdonohyperlink
1507     \let\@glstarget\@secondoftwo
1508   }
1509 }
1510 {}
```

`glsdonohyperlink` This command was only introduced in glossaries v4.20, so it may not be defined. For older glossaries versions, this won't be used if `hyperref` hasn't been loaded, which means the indexing will still take place.

```
1511 \def\glsdonohyperlink#1#2{{\glxtrprotectlinks #2}}
```

Reset `\@glslink` with patched versions:

```
1512 \ifcsundef{hyperlink}%
1513 {%
1514   \let\@glslink\glsdonohyperlink
1515 }%
1516 {%
1517   \let\@glslink\glsdohyperlink
1518 }
```

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

```
1519 \newcommand*{\glxtrprotectlinks}{%
1520   \KV@glslink@hyperfalse
1521   \KV@glslink@noindextrue
1522   \let\@gls@\@glxtrp@text@
1523   \let\@Gls@\@Glsxtrp@text@
1524   \let\@GLS@\@GLSxtrp@text@
1525   \let\@glspl@\@glxtrp@plural@
1526   \let\@Glspl@\@Glsxtrp@plural@
1527   \let\@GLSpl@\@GLSxtrp@plural@
1528   \let\@glxtrshort@\glxtrp@short@
```

```

1529 \let\@Glsxtrshort\@Glsxtrp@short@
1530 \let\@GLSxtrshort\@GLSxtrp@short@
1531 \let\@glsxtrlong\@glsxtrp@long@
1532 \let\@Glsxtrlong\@Glsxtrp@long@
1533 \let\@GLSxtrlong\@GLSxtrp@long@
1534 \let\@glsxtrshortpl\@glsxtrp@shortpl@
1535 \let\@Glsxtrshortpl\@Glsxtrp@shortpl@
1536 \let\@GLSxtrshortpl\@GLSxtrp@shortpl@
1537 \let\@glsxtrlongpl\@glsxtrp@longpl@
1538 \let\@Glsxtrlongpl\@Glsxtrp@longpl@
1539 \let\@GLSxtrlongpl\@GLSxtrp@longpl@
1540 \let\@acrshort\@glsxtrp@acrshort@
1541 \let\@Acrshort\@Glsxtrp@acrshort@
1542 \let\@ACRshort\@GLSxtrp@acrshort@
1543 \let\@acrshortpl\@glsxtrp@acrshortpl@
1544 \let\@Acrshortpl\@Glsxtrp@acrshortpl@
1545 \let\@ACRshortpl\@GLSxtrp@acrshortpl@
1546 \let\@acrlong\@glsxtrp@acrlong@
1547 \let\@Acrlong\@Glsxtrp@acrlong@
1548 \let\@ACRlong\@GLSxtrp@acrlong@
1549 \let\@acrlongpl\@glsxtrp@acrlongpl@
1550 \let\@Acrlongpl\@Glsxtrp@acrlongpl@
1551 \let\@ACRlongpl\@GLSxtrp@acrlongpl@
1552 }

```

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

@glsxtrp@text@

```
1553 \def\@glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@Glsxtrp@text@

```
1554 \def\@Glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtrp@text@

```
1555 \def\@GLSxtrp@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
1556 \def\@glsxtrp@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
1557 \def\@Glsxtrp@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

LSxtrp@plural@

```
1558 \def\@GLSxtrp@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glsxtrp@short@

```
1559 \def\@glsxtrp@short@#1#2[#3]{%
```

```
1560 {%
```



```

1561 \glsetabbrvfmt{\glscategory{#2}}%
1562 \glabbrvfont{\glentryshort{#2}}#3%
1563 }%
1564 }

```

Glsxtr@p@short@

```

1565 \def\@Glsxtr@p@short@#1#2[#3]{%
1566 {%
1567 \glsetabbrvfmt{\glscategory{#2}}%
1568 \glabbrvfont{\glentryshort{#2}}#3%
1569 }%
1570 }

```

GLSxtr@p@short@

```

1571 \def\@GLSxtr@p@short@#1#2[#3]{%
1572 {%
1573 \glsetabbrvfmt{\glscategory{#2}}%
1574 \mfirstucMakeUppercase{\glabbrvfont{\glentryshort{#2}}#3}%
1575 }%
1576 }

```

sxtr@p@shortpl@

```

1577 \def\@glsxtr@p@shortpl@#1#2[#3]{%
1578 {%
1579 \glsetabbrvfmt{\glscategory{#2}}%
1580 \glabbrvfont{\glentryshortpl{#2}}#3%
1581 }%
1582 }

```

sxtr@p@shortpl@

```

1583 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
1584 {%
1585 \glsetabbrvfmt{\glscategory{#2}}%
1586 \glabbrvfont{\glentryshortpl{#2}}#3%
1587 }%
1588 }

```

Sxtr@p@shortpl@

```

1589 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
1590 {%
1591 \glsetabbrvfmt{\glscategory{#2}}%
1592 \mfirstucMakeUppercase{\glabbrvfont{\glentryshortpl{#2}}#3}%
1593 }%
1594 }

```

@glsxtr@p@long@

```

1595 \def\@glsxtr@p@long@#1#2[#3]{\glentrylong{#2}#3}

```

```

@Glsxtr@p@long@
1596 \def\@Glsxtr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}}

@GLSxtr@p@long@
1597 \def\@GLSxtr@p@long@#1#2[#3]{%
1598   {\mfirstucMakeUppercase{\glslongfont{\Glsentrylong{#2}}#3}}}

lsxtr@p@longpl@
1599 \def\@glxtr@p@longpl@#1#2[#3]{\glsentrylongpl{#2}#3}}

lsxtr@p@longpl@
1600 \def\@Glsxtr@p@longpl@#1#2[#3]{\glslongfont{\Glsentrylongpl{#2}}#3}}

LSxtr@p@longpl@
1601 \def\@GLSxtr@p@longpl@#1#2[#3]{%
1602   {\mfirstucMakeUppercase{\glslongfont{\Glsentrylongpl{#2}}#3}}}

xtr@p@acrshort@
1603 \def\@glxtr@p@acrshort@#1#2[#3]{\acronymfont{\glsentryshort{#2}}#3}}

xtr@p@acrshort@
1604 \def\@Glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
1605 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
1606   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
1607 \def\@glxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
1608 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
1609 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
1610   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}

sxtr@p@acrlong@
1611 \def\@glxtr@p@acrlong@#1#2[#3]{\glsentrylong{#2}#3}}

sxtr@p@acrlong@
1612 \def\@Glsxtr@p@acrlong@#1#2[#3]{\Glsentrylong{#2}#3}}

Sxtr@p@acrlong@
1613 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
1614   {\mfirstucMakeUppercase{\glsentrylong{#2}#3}}}

```

tr@p@acrlongpl@

```
1615 \def\@glxstrp@acrlongpl@#1#2[#3]{\@glstrylongpl{#2}#3}}
```

tr@p@acrlongpl@

```
1616 \def\@Glsxtrp@acrlongpl@#1#2[#3]{\@Glsentrylongpl{#2}#3}}
```

tr@p@acrlongpl@

```
1617 \def\@GLSxtrp@acrlongpl@#1#2[#3]{%
1618 {\mfirstucMakeUppercase{\glstrylongpl{#2}#3}}}
```

Commands to minimise conflict.

\@glxstrp@opt

```
1619 \newcommand*{\@glxstrp@opt}{hyper=false,noindex}
```

\glxstrsetpopts Used in glossary to switch hyperlinks on for the \@glxstrp type of commands.

```
1620 \newcommand*{\glxstrsetpopts}[1]{%
1621 \renewcommand*{\@glxstrp@opt}{#1}%
1622 }
```

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

```
1623 \newcommand*{\lossxtrsetpopts}{%
1624 \glxstrsetpopts{noindex}%
1625 }
```

\@@glxstrp

```
1626 \newrobustcmd*{\@@glxstrp}[2]{%
```

Add scope.

```
1627 {%
1628 \let\glspostlinkhook\relax
1629 \csname#1\expandafter\endcsname\expandafter[\@glxstrp@opt]{#2}[]%
1630 }%
1631 }
```

\@glxstrp

```
1632 \newrobustcmd*{\@glxstrp}[2]{%
1633 \ifcsdef{gls#1}%
1634 {%
1635 \@glxstrp{gls#1}{#2}%
1636 }%
1637 {%
1638 \ifcsdef{glsxtr#1}%
1639 {%
1640 \@glxstrp{glsxtr#1}{#2}%
1641 }%
1642 {%
1643 \PackageError{glossaries-extra}{‘#1’ not recognised by
1644 \string\glxstrp}{%}
```

```

1645 }%
1646 }%
1647 }

```

\@Glsxtrp

```

1648 \newrobustcmd*{\@Glsxtrp}[2]{%
1649   \ifcsdef{Gls#1}%
1650   {%
1651     \@glsxtrp{Gls#1}{#2}%
1652   }%
1653   {%
1654     \ifcsdef{Glsxtr#1}%
1655     {%
1656       \@glsxtrp{Glsxtr#1}{#2}%
1657     }%
1658     {%
1659       \PackageError{glossaries-extra}{‘#1’ not recognised by
1660         \string\Glsxtrp}{}%
1661     }%
1662   }%
1663 }

```

\@GLSxtrp

```

1664 \newrobustcmd*{\@GLSxtrp}[2]{%
1665   \ifcsdef{GLS#1}%
1666   {%
1667     \@glsxtrp{GLS#1}{#2}%
1668   }%
1669   {%
1670     \ifcsdef{GLSxtr#1}%
1671     {%
1672       \@glsxtrp{GLSxtr#1}{#2}%
1673     }%
1674     {%
1675       \PackageError{glossaries-extra}{‘#1’ not recognised by
1676         \string\GLSxtrp}{}%
1677     }%
1678   }%
1679 }

```

\glsxtr@entry@p

```

1680 \newrobustcmd*{\glsxtr@headentry@p}[2]{%
1681   \glsifattribute{#1}{headuc}{true}%
1682   {%
1683     \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
1684   }%
1685   {%
1686     \@gls@entry@field{#1}{#2}%
1687   }%

```

1688 }

\glsxtrp Not robust as it needs to expand somewhat.

```
1689 \ifdef\teorpdfstring
1690 {
1691   \newcommand{\glsxtrp}[2]{%
1692     \protect\NoCaseChange
1693     {%
1694       \protect\teorpdfstring
1695       {%
1696         \protect\glsxtrifinmark
1697         {%
1698           \ifcsdef{glsxtrhead#1}%
1699           {%
1700             {\protect\csuse{glsxtrhead#1}{#2}}%
1701             }%
1702             {%
1703               \glsxtr@headentry@p{#2}{#1}%
1704               }%
1705             }%
1706             {%
1707               \@glsxtrp{#1}{#2}%
1708               }%
1709             }%
1710             {%
1711               \protect\@gls@entry@field{#2}{#1}%
1712               }%
1713             }%
1714           }
1715 }
1716 {
1717   \newcommand{\glsxtrp}[2]{%
1718     \protect\NoCaseChange
1719     {%
1720       \protect\glsxtrifinmark
1721       {%
1722         \ifcsdef{glsxtrhead#1}%
1723         {%
1724           {\protect\csuse{glsxtrhead#1}}%
1725           }%
1726           {%
1727             \glsxtr@headentry@p{#2}{#1}%
1728             }%
1729           }%
1730           {%
1731             \@glsxtrp{#1}{#2}%
1732             }%
1733           }%
1734         }
```

1735 }

Provide short synonyms for the most common option.

`\glsps`

```
1736 \newcommand*{\glsps}{\glsxtrp{short}}
```

`\glspt`

```
1737 \newcommand*{\glspt}{\glsxtrp{text}}
```

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

```
1738 \ifdef\texorpdfstring
1739 {
1740   \newcommand{\Glsxtrp}[2]{%
1741     \protect\NoCaseChange
1742     {%
1743       \protect\texorpdfstring
1744       {%
1745         \protect\glsxtrifinmark
1746         {%
1747           \ifcsdef{Glsxtrhead#1}%
1748             {%
1749               {\protect\csuse{Glsxtrhead#1}{#2}}%
1750             }%
1751             {%
1752               \protect\@Gls@entry@field{#2}{#1}%
1753             }%
1754             }%
1755             {%
1756               \@Glsxtrp{#1}{#2}%
1757             }%
1758             }%
1759             {%
1760               \protect\@gls@entry@field{#2}{#1}%
1761             }%
1762             }%
1763         }
1764     }
1765 {
1766   \newcommand{\Glsxtrp}[2]{%
1767     \protect\NoCaseChange
1768     {%
1769       \protect\glsxtrifinmark
1770       {%
1771         \ifcsdef{Glsxtrhead#1}%
1772           {%
1773             {\protect\csuse{Glsxtrhead#1}}%
1774           }%

```

```

1775      {%
1776      \protect\@Gls@entry@field{#2}{#1}%
1777      }%
1778    }%
1779    {%
1780    \@Glsxtrp{#1}{#2}%
1781    }%
1782  }%
1783 }
1784 }

```

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

```

1785 \ifdef\teorpdfstring
1786 {
1787   \newcommand{\GLSxtrp}[2]{%
1788     \protect\NoCaseChange
1789     {%
1790       \protect\teorpdfstring
1791       {%
1792         \protect\glsxtrifinmark
1793         {%
1794           \ifcsdef{GLSxtr#1}%
1795           {%
1796             {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
1797           }%
1798           {%
1799             \protect\mfirstucMakeUppercase
1800             {%
1801               \protect\@gls@entry@field{#2}{#1}%
1802             }%
1803           }%
1804         }%
1805       }%
1806       \@GLSxtrp{#1}{#2}%
1807     }%
1808   }%
1809   {%
1810     \protect\@gls@entry@field{#2}{#1}%
1811   }%
1812 }%
1813 }
1814 }
1815 {
1816   \newcommand{\GLSxtrp}[2]{%
1817     \protect\NoCaseChange
1818     {%
1819       \protect\glsxtrifinmark
1820       {%
1821         \ifcsdef{GLSxtr#1}%

```

```

1822      {%
1823      {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
1824      }%
1825      {%
1826      \protect\mfirstucMakeUppercase
1827      {%
1828      \protect\@gls@entry@field{#2}{#1}%
1829      }%
1830      }%
1831      }%
1832      {%
1833      \@GLSxtrp{#1}{#2}%
1834      }%
1835      }%
1836    }
1837 }

```

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.

```

1838 \renewcommand*{\@glsunset}[1]{%
1839   \@@glsunset{#1}%
1840   \glsxtrpostunset{#1}%
1841 }%

```

glsxtrpostunset

```

1842 \newcommand*{\glsxtrpostunset}[1]{%

```

\@glslocalunset Local unset.

```

1843 \renewcommand*{\@glslocalunset}[1]{%
1844   \@@glslocalunset{#1}%
1845   \glsxtrpostlocalunset{#1}%
1846 }%

```

rpostlocalunset

```

1847 \newcommand*{\glsxtrpostlocalunset}[1]{%

```

\@glsreset Global reset.

```

1848 \renewcommand*{\@glsreset}[1]{%
1849   \@@glsreset{#1}%
1850   \glsxtrpostreset{#1}%
1851 }%

```


glsxtrpostreset

```
1852 \newcommand*{\glsxtrpostreset}[1]{}
```

\@glslocalreset Local reset.

```
1853 \renewcommand*{\@glslocalreset}[1]{%
1854   \@glslocalreset{#1}%
1855   \glsxtrpostlocalreset{#1}%
1856 }%
```

rpostlocalreset

```
1857 \newcommand*{\glsxtrpostlocalreset}[1]{}
```

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

```
1858 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

Enable entry counting:

```
1859   \glsenableentrycount
```

Redefine \gls etc:

```
1860   \renewcommand*{\gls}{\cglsl}%
1861   \renewcommand*{\Gls}{\cGls}%
1862   \renewcommand*{\glspl}{\cglspl}%
1863   \renewcommand*{\Glspl}{\cGlspl}%
1864   \renewcommand*{\GLS}{\cGLS}%
1865   \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
1866   \@glsxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
1867   \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
1868   \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
1869     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
1870       can't be used with \string\GlsXtrEnableEntryCounting}%
1871     {Use one or other but not both commands}}%
1872 }
```

ycountunsetattr

```
1873 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
1874   \@for\@glsxtr@cat:=#1\do
1875   {%
1876     \ifdefempty{\@glsxtr@cat}{}%
1877     {%
1878       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
1879     }%
1880   }%
1881 }
```

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

nableentrycount

```
1882 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
1883 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}{%
```

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

```
1884 \renewcommand*{\gls@defdocnewglossaryentry}{%
1885 \renewcommand*\newglossaryentry[2]{%
1886 \PackageError{glossaries}{\string\newglossaryentry\space
1887 may only be used in the preamble when entry counting has
1888 been activated}{If you use \string\glsenableentrycount\space
1889 you must place all entry definitions in the preamble not in
1890 the document environment}}%
1891 }%
1892 }%
```

New commands to access new fields:

```
1893 \newcommand*{\glsentrycurrcount}[1]{%
1894 \ifcsundef{glo@\glsdetoklabel{##1}@currcount}}%
1895 {0}{\@gls@entry@field{##1}{currcount}}}%
1896 }%
1897 \newcommand*{\glsentryprevcount}[1]{%
1898 \ifcsundef{glo@\glsdetoklabel{##1}@prevcount}}%
1899 {0}{\@gls@entry@field{##1}{prevcount}}}%
1900 }%
```

Adjust post unset and reset:

```
1901 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
1902 \renewcommand*{\glsxtrpostunset}[1]{%
1903 \@glsxtr@entrycount@org@unset{##1}%
1904 \@gls@increment@currcount{##1}%
1905 }%
1906 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
1907 \renewcommand*{\glsxtrpostlocalunset}[1]{%
1908 \@glsxtr@entrycount@org@localunset{##1}%
1909 \@gls@local@increment@currcount{##1}%
1910 }%
1911 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
1912 \renewcommand*{\glsxtrpostreset}[1]{%
1913 \@glsxtr@entrycount@org@reset{##1}%
1914 \csgdef{glo@\glsdetoklabel{##1}@currcount}{0}%
1915 }%
1916 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
1917 \renewcommand*{\glsxtrpostlocalreset}[1]{%
1918 \@glsxtr@entrycount@org@localreset{##1}%
1919 \csdef{glo@\glsdetoklabel{##1}@currcount}{0}%
1920 }%
```

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

```

1921 \let\@cglS@\@cglS@
1922 \let\@cglSpl@\@cglSpl@
1923 \let\@cGLS@\@cGLS@
1924 \let\@cGlSpl@\@cGlSpl@
1925 \let\@cGLS@\@cGLS@
1926 \let\@cGLSpl@\@cGLSpl@

```

The rest is as the original definition.

```

1927 \AtEndDocument{\@glS@write@entrycounts}%
1928 \renewcommand*{\@glS@entry@count}[2]{%
1929   \csgdef{glo@\glSdetoklabel{##1}@prevcount}{##2}%
1930 }%
1931 \let\glSenableentrycount\relax
1932 \renewcommand*{\glSenableentryunitcount}{%
1933   \PackageError{glossaries-extra}{\string\glSenableentryunitcount\space
1934     can't be used with \string\glSenableentrycount}%
1935   {Use one or other but not both commands}%
1936 }%
1937 }

```

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

```

1938 \renewcommand*{\@glS@write@entrycounts}{%
1939   \immediate\write\@auxout
1940     {\string\providecommand*{\string\@glS@entry@count}[2]{}}%
1941   \count@=0\relax
1942   \forallglSentries{\@glSentry}{%
1943     \glShasattribute{\@glSentry}{entrycount}%
1944     {%
1945       \ifglSused{\@glSentry}%
1946       {%
1947         \immediate\write\@auxout
1948           {\string\@glS@entry@count{\@glSentry}{\glSentrycurrcount{\@glSentry}}}%
1949       }%
1950     }%
1951     \advance\count@ by \@ne
1952   }%
1953 }%
1954 }%
1955 \ifnum\count@=0
1956   \GlossariesExtraWarningNoLine{Entry counting has been enabled
1957     \MessageBreak with \string\glSenableentrycount\space but the
1958     \MessageBreak attribute 'entrycount' hasn't
1959     \MessageBreak been assigned to any of the defined
1960     \MessageBreak entries}%
1961 \fi
1962 }

```

trifcounttrigger

<code>\glxtrifcounttrigger{<label>}{<trigger format>}{<normal>}</code>
--

```
1963 \newcommand*{\glxtrifcounttrigger}[3]{%
1964   \glshasattribute{#1}{entrycount}%
1965   {%
1966     \ifnum\gl Sentryprevcount{#1}>\glsetattribute{#1}{entrycount}\relax
1967     #3%
1968   \else
1969     #2%
1970   \fi
1971 }%
1972 {#3}%
1973 }
```

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

\@@cgl@

```
1974 \def\@@cgl@#1#2[#3]{%
1975   \glxtrifcounttrigger{#2}%
1976   {%
1977     \cglformat{#2}{#3}%
1978     \gl sunset{#2}%
1979   }%
1980   {%
1981     \@gls@{#1}{#2}[#3]%
1982   }%
1983 }
```

\@@cgl@

```
1984 \def\@@cgl@#1#2[#3]{%
1985   \glxtrifcounttrigger{#2}%
1986   {%
1987     \cgl@format{#2}{#3}%
1988     \gl sunset{#2}%
1989   }%
1990   {%
1991     \@gls@{#1}{#2}[#3]%
1992   }%
1993 }
```

\@@cGls@

```
1994 \def\@@cGls@#1#2[#3]{%
1995   \glxtrifcounttrigger{#2}%
1996   {%
1997     \cGlsformat{#2}{#3}%
1998     \gl sunset{#2}%

```

```

1999 }%
2000 {%
2001 \cGls@{#1}{#2}[#3]%
2002 }%
2003 }%

```

\cGlspl@

```

2004 \def\cGlspl@#1#2[#3]{%
2005 \glstrifcounttrigger{#2}%
2006 {%
2007 \cGlsplformat{#2}{#3}%
2008 \glunset{#2}%
2009 }%
2010 {%
2011 \cGlspl@{#1}{#2}[#3]%
2012 }%
2013 }%

```

\cGLS@

```

2014 \def\cGLS@#1#2[#3]{%
2015 \glstrifcounttrigger{#2}%
2016 {%
2017 \cGLSformat{#2}{#3}%
2018 \glunset{#2}%
2019 }%
2020 {%
2021 \cGLS@{#1}{#2}[#3]%
2022 }%
2023 }%

```

\cGLSpl@

```

2024 \def\cGLSpl@#1#2[#3]{%
2025 \glstrifcounttrigger{#2}%
2026 {%
2027 \cGLSplformat{#2}{#3}%
2028 \glunset{#2}%
2029 }%
2030 {%
2031 \cGLSpl@{#1}{#2}[#3]%
2032 }%
2033 }%
2034 %
2035 % Remove default warnings from \cs{cgl} etc so that it can be used
2036 % interchangeably with \cs{gl} etc.
2037 %\begin{macro}\cgl@
2038 % \begin{macrocode}
2039 \def\cgl@#1#2[#3]{\cgl@{#1}{#2}[#3]}

```

\cGls@

```
2040 \def\@cGLs@#1#2[#3]{\@GLs@{#1}{#2}[#3]}
```

```
\@cglsp1@
```

```
2041 \def\@cglsp1@#1#2[#3]{\@glsp1@{#1}{#2}[#3]}
```

```
\@cGLsp1@
```

```
2042 \def\@cGLsp1@#1#2[#3]{\@GLsp1@{#1}{#2}[#3]}
```

Add all upper case versions not provided by glossaries.

```
\cGLS
```

```
2043 \newrobustcmd*{\cGLS}{\@glshyp@opt\@cGLS}
```

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

```
2044 \newcommand*{\@cGLS}[2][\%
```

```
2045 \new@ifnextchar[\@cGLS@{#1}{#2}]{\@cGLS@{#1}{#2}[ ]}%
```

```
2046 }
```

```
\@cGLS@
```

```
2047 \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.

```
2048 \newcommand*{\cGLSformat}[2]{\%
```

```
2049 \expandafter\mfirstucMakeUppercase\expandafter{\cglformat{#1}{#2}}%
```

```
2050 }
```

```
\cGLSp1
```

```
2051 \newrobustcmd*{\cGLSp1}{\@glshyp@opt\@cGLSp1}
```

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

```
2052 \newcommand*{\@cGLSp1}[2][\%
```

```
2053 \new@ifnextchar[\@cGLSp1@{#1}{#2}]{\@cGLSp1@{#1}{#2}[ ]}%
```

```
2054 }
```

```
\@cGLSp1@
```

```
2055 \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.

```
2056 \newcommand*{\cGLSplformat}[2]{\%
```

```
2057 \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
```

```
2058 }
```

Modify the trigger formats to check for the regular attribute.

\cglsformat

```
2059 \renewcommand*{\cglsformat}[2]{%
2060   \glsifregular{#1}
2061   {\glsentryfirst{#1}}%
2062   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}#2%
2063 }
```

\cGlsformat

```
2064 \renewcommand*{\cGlsformat}[2]{%
2065   \glsifregular{#1}
2066   {\Glsentryfirst{#1}}%
2067   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}#2%
2068 }
```

\cglsplformat

```
2069 \renewcommand*{\cglsplformat}[2]{%
2070   \glsifregular{#1}
2071   {\glsentryfirstplural{#1}}%
2072   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}#2%
2073 }
```

\cGlsplformat

```
2074 \renewcommand*{\cGlsplformat}[2]{%
2075   \glsifregular{#1}
2076   {\Glsentryfirstplural{#1}}%
2077   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2%
2078 }
```

New code similar to above for unit counting.

defunitcounters

```
2079 \newcommand*{\@@newglossaryentry@defunitcounters}{%
2080   \edef\@glo@countunit{\csuse{@glxtr@categoryattr@@\@glo@category @unitcount}}%
2081   \ifdefvoid\@glo@countunit
2082   {}%
2083   {%
2084     \@glxtr@ifunitcounter{\@glo@countunit}%
2085     {}%
2086     {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}%
2087   }%
2088 }
```

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

```
2089 \newcommand*{\@glxtr@unitcountlist}{}
```

@addunitcounter

```
2090 \newcommand*{\@glxtr@addunitcounter}[1]{%
2091   \listadd{\@glxtr@unitcountlist}{#1}%

```

```

2092 \ifcsundef{glxstr@theunit@#1}
2093 {%
2094   \ifcsdef{theH#1}%
2095   {\csdef{glxstr@theunit@#1}{\csuse{theH#1}}}%
2096   {\csdef{glxstr@theunit@#1}{\csuse{the#1}}}%
2097 }%
2098 {}%
2099 }

```

r@ifunitcounter

```

2100 \newcommand*{\@glxstr@ifunitcounter}[3]{%
2101   \xifinlist{#1}{\@glxstr@unitcountlist}{#2}{#3}%
2102 }

```

urrentunitcount

```

2103 \newcommand*\@glxstr@currentunitcount[1]{%
2104   glo@\glsdetoklabel{#1}@currunit@\glsggetattribute{#1}{unitcount}.%
2105   \csuse{glxstr@theunit@\glsggetattribute{#1}{unitcount}}%
2106 }

```

eviousunitcount

```

2107 \newcommand*\@glxstr@previousunitcount[1]{%
2108   glo@\glsdetoklabel{#1}@prevunit@\glsggetattribute{#1}{unitcount}.%
2109   \csuse{glxstr@theunit@\glsggetattribute{#1}{unitcount}}%
2110 }

```

t@currunitcount

```

2111 \newcommand*{\@glx@increment@currunitcount}[1]{%
2112   \glshasattribute{#1}{unitcount}%
2113   {%
2114     \edef\@glxstr@csname{\@glxstr@currentunitcount{#1}}%
2115     \ifcsundef{\@glxstr@csname}%
2116     {%
2117       \csgdef{\@glxstr@csname}{1}%
2118       \listcsxadd
2119       {glo@\glsdetoklabel{#1}@unitlist}%
2120       {\glsggetattribute{#1}{unitcount}.%
2121        \csuse{glxstr@theunit@\glsggetattribute{#1}{unitcount}}%
2122       }%
2123     }%
2124     {%
2125       \csxdef{\@glxstr@csname}%
2126       {\number\numexpr\csname\@glxstr@csname\endcsname+1}%
2127     }%
2128   }%
2129   {}%
2130 }

```

t@currunitcount


```

2131 \newcommand*{\@gls@local@increment@currunitcount}[1]{%
2132   \gls@hasattribute{#1}{unitcount}%
2133   {%
2134     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
2135     \ifcsundef{\@glsxtr@csname}%
2136     {%
2137       \csdef{\@glsxtr@csname}{1}%
2138       \listcseadd
2139         {glo@\glsdetoklabel{#1}@unitlist}%
2140         {\glsgetattribute{#1}{unitcount}.%
2141          \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
2142       }%
2143     }%
2144     {%
2145       \csedef{\@glsxtr@csname}%
2146         {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
2147       }%
2148     }%
2149   }%
2150 }

```

r@currunitcount

```

2151 \newcommand*{\@glsxtr@currunitcount}[2]{%
2152   \ifcsundef
2153     {glo@\glsdetoklabel{#1}@currunit@#2}%
2154     {0}%
2155     {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}%
2156 }%

```

r@prevunitcount

```

2157 \newcommand*{\@glsxtr@prevunitcount}[2]{%
2158   \ifcsundef
2159     {glo@\glsdetoklabel{#1}@prevunit@#2}%
2160     {0}%
2161     {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}%
2162 }%

```

eentryunitcount

```

2163 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
2164   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
2165   \renewcommand*{\gls@defdocnewglossaryentry}{%
2166     \renewcommand*\newglossaryentry[2]{%
2167       \PackageError{glossaries}{\string\newglossaryentry\space
2168         may only be used in the preamble when entry counting has
2169         been activated}{If you use \string\glsenableentryunitcount\space
2170         you must place all entry definitions in the preamble not in

```

```

2171     the document environment}%
2172 }%
2173 }%

```

New commands to access new fields:

```

2174 \newcommand*{\glentrycurrcount}[1]{%
2175   \@glxtr@currunitcount{##1}{\glsetattribute{##1}{unitcount}.%
2176   \csuse{glxtr@theunit@\glsetattribute{##1}{unitcount}}}%
2177 }%
2178 \newcommand*{\glentryprevcount}[1]{%
2179   \@glxtr@prevunitcount{##1}{\glsetattribute{##1}{unitcount}.%
2180   \csuse{glxtr@theunit@\glsetattribute{##1}{unitcount}}}%
2181 }%

```

Access total count:

```

2182 \newcommand*{\glentryprevtotalcount}[1]{%
2183   \ifcsundef{glo@\glsetoklabel{##1}@prevunittotal}%
2184   {0}%
2185   {%
2186     \number\csuse{glo@\glsetoklabel{##1}@prevunittotal}
2187   }%
2188 }%

```

Access max value:

```

2189 \newcommand*{\glentryprevmaxcount}[1]{%
2190   \ifcsundef{glo@\glsetoklabel{##1}@prevunitmax}%
2191   {0}%
2192   {%
2193     \number\csuse{glo@\glsetoklabel{##1}@prevunitmax}
2194   }%
2195 }%

```

Adjust post unset and reset:

```

2196 \let\@glxtr@entryunitcount@org@unset\glxtrpostunset
2197 \renewcommand*{\glxtrpostunset}[1]{%
2198   \@glxtr@entryunitcount@org@unset{##1}%
2199   \@gl@increment@currunitcount{##1}%
2200 }%
2201 \let\@glxtr@entryunitcount@org@localunset\glxtrpostlocalunset
2202 \renewcommand*{\glxtrpostlocalunset}[1]{%
2203   \@glxtr@entryunitcount@org@localunset{##1}%
2204   \@gl@local@increment@currunitcount{##1}%
2205 }%
2206 \let\@glxtr@entryunitcount@org@reset\glxtrpostreset
2207 \renewcommand*{\glxtrpostreset}[1]{%
2208   \glshasattribute{##1}{unitcount}%
2209   {%
2210     \edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
2211     \ifcsundef{\@glxtr@csname}%
2212     {}%
2213     {\csgdef{\@glxtr@csname}{0}}%

```

```

2214 }%
2215 {}%
2216 }%
2217 \let\@glstr@entryunitcount@org@localreset\glstrpostlocalreset
2218 \renewcommand*{\glstrpostlocalreset}[1]{%
2219   \@glstr@entryunitcount@org@localreset{##1}%
2220   \glshasattribute{##1}{unitcount}%
2221   {%
2222     \edef\@glstr@csname{\@glstr@currentunitcount{##1}}%
2223     \ifcsundef{\@glstr@csname}%
2224     {}%
2225     {\csdef{\@glstr@csname}{0}}%
2226   }%
2227   {}%
2228 }%

```

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

```

2229 \let\@cgl@\@cgl@
2230 \let\@cgl@sp1\@cgl@sp1
2231 \let\@cGLS\@cGLS
2232 \let\@cGL@sp1\@cGL@sp1
2233 \let\@cGLS\@cGLS
2234 \let\@cGL@sp1\@cGL@sp1

```

Write information to the aux file.

```

2235 \AtEndDocument{\@gl@write@entryunitcounts}%
2236 \renewcommand*{\@gl@entry@unitcount}[3]{%
2237   \csgdef{glo@gl@detoklabel{##1}@prevunit@##3}{##2}%
2238   \ifcsundef{glo@gl@detoklabel{##1}@prevunittotal}%
2239   {\csgdef{glo@gl@detoklabel{##1}@prevunittotal}{##2}}%
2240   {%
2241     \csxdef{glo@gl@detoklabel{##1}@prevunittotal}{
2242       \number\numexpr\csuse{glo@gl@detoklabel{##1}@prevunittotal}+##2}%
2243     }%
2244     \ifcsundef{glo@gl@detoklabel{##1}@prevunitmax}%
2245     {\csgdef{glo@gl@detoklabel{##1}@prevunitmax}{##2}}%
2246     {%
2247       \ifnum\csuse{glo@gl@detoklabel{##1}@prevunitmax}<##2
2248       \csgdef{glo@gl@detoklabel{##1}@prevunitmax}{##2}%
2249       \fi
2250     }%
2251   }%
2252 \let\gl@enableentryunitcount\relax
2253 \renewcommand*{\gl@enableentrycount}{%
2254   \PackageError{glossaries-extra}{\string\gl@enableentrycount\space
2255     can't be used with \string\gl@enableentryunitcount}%
2256   {Use one or other but not both commands}%
2257 }%
2258 }

```

```

2259 \@onlypreamble\glsenableentryunitcount

entry@unitcount
2260 \newcommand*{\@gls@entry@unitcount}[3]{%

ryunitcounts@do
2261 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
2262   \immediate\write\@auxout
2263   {\string\@gls@entry@unitcount
2264     {\@glsentry}%
2265     {\@glsxtr@currunitcount{\@glsentry}{#1}%
2266     }%
2267     {#1}}%
2268 }

entryunitcounts
2269 \newcommand*{\@gls@write@entryunitcounts}{%
2270   \immediate\write\@auxout
2271   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
2272   \count@=0\relax
2273   \forallglsentries{\@glsentry}{%
2274     \glsattribute{\@glsentry}{unitcount}%
2275     {%
2276       \ifglsused{\@glsentry}%
2277       {%
2278         \forlistcsloop
2279           {\@gls@write@entryunitcounts@do}%
2280           {glo@\glsdetoklabel{\@glsentry}@unitlist}%
2281       }%
2282     }%
2283     \advance\count@ by \@ne
2284   }%
2285 }%
2286 }%
2287 \ifnum\count@=0
2288   \GlossariesExtraWarningNoLine{Entry counting has been enabled
2289     \MessageBreak with \string\glsenableentryunitcount\space but the
2290     \MessageBreak attribute ‘unitcount’ hasn’t
2291     \MessageBreak been assigned to any of the defined
2292     \MessageBreak entries}%
2293 \fi
2294 }

```

entryUnitCounting 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.

```

2295 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

  Enable entry counting:

2296   \glsenableentryunitcount

```

Redefine `\gls` etc:

```
2297 \renewcommand*{\gls}{\cgl{s}}%
2298 \renewcommand*{\Gls}{\cGls}%
2299 \renewcommand*{\glspl}{\cgl{s}pl}%
2300 \renewcommand*{\Glspl}{\cGlspl}%
2301 \renewcommand*{\GLS}{\cGLS}%
2302 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2303 \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%
```

In case this command is used again:

```
2304 \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
2305 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
2306   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
2307     can't be used with \string\GlsXtrEnableEntryUnitCounting}%
2308   {Use one or other but not both commands}}%
2309 }
```

`tcountunsetattr`

```
2310 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
2311   \@for\@glsxtr@cat:=#1\do
2312   {%
2313     \ifdefempty{\@glsxtr@cat}{}%
2314     {%
2315       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
2316       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
2317     }%
2318   }%
2319 }
```

1.3.6 Acronym Modifications

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

`nericNewAcronym`

```
2320 \renewcommand*{\SetGenericNewAcronym}{%
2321   \let\@Gls@entryname\@Gls@acrenryname
2322   \renewcommand{\newacronym}[4][ ]{%
2323     \ifdefempty{\@glsacronymlists}%
2324     {%
2325       \def\@glo@type{\acronymtype}%
2326       \setkeys{glossentry}{##1}%
2327       \DeclareAcronymList{\@glo@type}%
2328     }%
2329   }%
```

```

2328 }%
2329 {}%
2330 \glskeylisttok{##1}%
2331 \glslabeltok{##2}%
2332 \glsshorttok{##3}%
2333 \glslongtok{##4}%
2334 \newacronymhook
2335 \protected@edef\@do@newglossaryentry{%
2336   \noexpand\newglossaryentry{\the\glslabeltok}%
2337   {%
2338     type=\acronymtype,%
2339     name={\expandonce{\acronymentry{##2}}},%
2340     sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
2341     text={\the\glsshorttok},%
2342     short={\the\glsshorttok},%
2343     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
2344     long={\the\glslongtok},%
2345     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
2346     category=acronym,
2347     \GenericAcronymFields,%
2348     \the\glskeylisttok
2349   }%
2350 }%
2351 \@do@newglossaryentry
2352 }%
2353 \renewcommand*{\acrfullfmt}[3]{%
2354   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
2355 \renewcommand*{\Acrfullfmt}[3]{%
2356   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
2357 \renewcommand*{\ACRfullfmt}[3]{%
2358   \glslink[##1]{##2}{%
2359     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
2360 \renewcommand*{\acrfullplfmt}[3]{%
2361   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
2362 \renewcommand*{\Acrfullplfmt}[3]{%
2363   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
2364 \renewcommand*{\ACRfullplfmt}[3]{%
2365   \glslink[##1]{##2}{%
2366     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
2367 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
2368 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
2369 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
2370 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
2371 }

```

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:

```

2372 \let\@glsxtr@org@setacronymstyle\setacronymstyle

```

2373 \let\@glxtr@org@newacronymstyle\newacronymstyle

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

```

2374 \newcommand*\MakeAcronymsAbbreviations{%
2375   \renewcommand*\newacronym}[4][{}]{%
2376     \newabbreviation[type=\acronymtype,category=acronym,##1]{##2}{##3}{##4}%
2377   }%
2378   \renewcommand*\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
2379   \renewcommand*\acronymfont}[1]{\glsabbrvfont{##1}}%
2380   \renewcommand*\setacronymstyle}[1]{%
2381     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}
2382     unavailable.
2383     Use \string\setabbreviationstyle\space instead.
2384     The original acronym interface can be restored with
2385     \string\RestoreAcronyms}%
2386   }%
2387   \renewcommand*\newacronymstyle}[1]{%
2388     \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
2389     available unless you restore the original acronym interface with
2390     \string\RestoreAcronyms}%
2391     \@glxtr@org@newacronymstyle{##1}%
2392   }%
2393 }

```

Switch acronyms to abbreviations:

2394 \MakeAcronymsAbbreviations

RestoreAcronyms Restore acronyms to glossaries interface.

```

2395 \newcommand*\RestoreAcronyms{%
2396   \SetGenericNewAcronym
2397   \renewcommand*\firstacronymfont}[1]{\acronymfont{##1}}%
2398   \renewcommand*\acronymfont}[1]{##1}%
2399   \let\setacronymstyle\@glxtr@org@setacronymstyle
2400   \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.

```

2401 \renewcommand*\@gls@link@checkfirsthyper{%
2402   \ifglused{\glslabel}%
2403   {\let\glxtrifwasfirstuse\@secondoftwo}
2404   {\let\glxtrifwasfirstuse\@firstoftwo}%
2405   \@glxtr@org@checkfirsthyper
2406 }
2407 \glssetcategoryattribute{acronym}{regular}{false}%
2408 \setacronymstyle{long-short}%
2409 }

```

```

\glsacspace Allow the user to customise the maximum value.
2410 \renewcommand*{\glsacspace}[1]{%
2411   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
2412   \ifdim\dimen@<\glsacspacemax~\else\space\fi
2413 }

```

```

\glsacspacemax Value used in the above.
2414 \newcommand*{\glsacspacemax}{3em}

```

1.3.7 Indexing and Displaying Glossaries

From time-to-time users ask if they can have one glossary sorted normally and another sorted by definition or usage. With the base glossaries package this can only be achieved with the “noidx” commands (Option 1). This is an attempt to mix and match.

First we need a list of the glossaries that require `makeindex/xindy`.

```

r@reg@glosslist
2415 \newcommand*{\@glsxtr@reg@glosslist}{}

```

Save the original definition of `\makeglossaries`:

```

2416 \let\@glsxtr@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
2417 \renewcommand*{\makeglossaries}[1] [] {%
2418   \ifblank{#1}%
2419   {\@glsxtr@org@makeglossaries}%
2420   {%
2421     \edef\@glsxtr@reg@glosslist{#1}%
2422     \ifundef{\glswrite}{\newwrite\glswrite}{}%
2423     \protected@write\@auxout{}\string\providecommand
2424       \string\@glsorder[1]{}}
2425     \protected@write\@auxout{}\string\providecommand
2426       \string\@istfilename[1]{}}
2427     \protected@write\@auxout{}\string\@istfilename{\istfilename}}%
2428     \protected@write\@auxout{}\string\@glsorder{\glsorder}}
2429     \write\@auxout{\string\providecommand\string\@gls@reference[3]{}}%

```

Iterate through each supplied glossary type and activate it.

```

2430   \@for\@glo@type:=#1\do{%
2431     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
2432   }%

```

New glossaries must be created before `\makeglossaries`:

```

2433 \renewcommand*{\newglossary}[4] [] {%
2434 \PackageError{glossaries}{New glossaries

```



```

2435 must be created before \string\makeglossaries}{You need
2436 to move \string\makeglossaries\space after all your
2437 \string\newglossary\space commands}}%

```

Any subsequence instances of this command should have no effect

```

2438 \let\@makeglossary\relax
2439 \let\makeglossary\relax
2440 \let\makeglossaries\relax

```

Disable all commands that have no effect after \makeglossaries

```

2441 \@disable@onlypremakeg

```

Allow see key:

```

2442 \let\gls@checkseeallowed\relax

```

Suppress warning about no \makeglossaries

```

2443 \let\warn@nomakeglossaries\relax
2444 \def\warn@noprintglossary{%
2445   \GlossariesWarningNoLine{No \string\printglossary\space
2446     or \string\printglossaries\space
2447     found.^^J(Remove \string\makeglossaries\space if you don't want
2448     any glossaries.)^^JThis document will not have a glossary}%
2449 }%

```

Adjust display number list to check for type:

```

2450 \renewcommand*{\glsdisplaynumberlist}[1]{%
2451   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2452   {\@glsxtr@idx@displaynumberlist{##1}}%
2453   {\@glsxtr@noidx@displaynumberlist{##1}}%
2454 }%

```

Adjust entry list:

```

2455 \renewcommand*{\glsentrynumberlist}[1]{%
2456   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2457   {\@glsxtr@idx@entrynumberlist{##1}}%
2458   {\@glsxtr@noidx@entrynumberlist{##1}}%
2459 }%

```

Adjust number list loop

```

2460 \renewcommand*{\glsnumberlistloop}[2]{%
2461   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2462   {%
2463     \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
2464       not available for glossary '##1'}{ }%
2465   }%
2466   {\@glsxtr@noidx@numberlistloop{##1}{##2}}%
2467 }%

```

Only sanitize sort for normal indexing glossaries.

```

2468 \renewcommand*{\glsprestandardsort}[3]{%
2469   \expandafter\DTLifinlist\expandafter{##2}{\@glsxtr@reg@glosslist}%
2470   {%

```

```

2471     \glsdosanitizesort
2472 }%
2473 {%
2474     \ifglssanitizesort
2475     \@gls@noidx@sanitizesort
2476     \else
2477     \@gls@noidx@nosanitizesort
2478     \fi
2479 }%
2480 }%

```

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

```

2481 \renewcommand*\new@glossaryentry[2]{%
2482   \PackageError{glossaries-extra}{Glossary entries must be defined
2483     in the preamble\MessageBreak when you use the optional argument
2484     of \string\makeglossaries}{Either move your definitions to the
2485     preamble or don't use the optional argument of
2486     \string\makeglossaries}%
2487 }%

```

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).

```

2488 \renewcommand*\@printgloss@setsort{%
2489   \renewcommand*\@glo@assign@sortkey{%
2490     \edef\@glo@type{\@glo@type}%
2491     \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glsxtr@reg@glosslist}%
2492     {%
2493       \@@glo@no@assign@sortkey
2494     }%
2495     {%
2496       \@@glo@assign@sortkey
2497     }%
2498   }%
2499   \def\@glo@sorttype{\@glo@default@sorttype}%
2500 }%

```

Check automake setting:

```

2501 \ifglsautomake
2502   \renewcommand*\@gls@doautomake{%
2503     \@for\@gls@type:=\@glsxtr@reg@glosslist\do{%
2504       \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
2505     }%
2506   }%
2507 \fi
2508 }%
2509 }

```

Display number list for the regular version:

`splaynumberlist`

2510 \let\@glxtr@idx@displaynumberlist\glsdisplaynumberlist

Display number list for the “noidx” version:

splaynumberlist

```

2511 \newcommand*{\@glxtr@noidx@displaynumberlist}[1]{%
2512   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
2513   \ifdef\@gls@loclist
2514   {%
2515     \def\@gls@noidx@loclist@sep{%
2516       \def\@gls@noidx@loclist@sep{%
2517         \def\@gls@noidx@loclist@sep{%
2518           \glsnumlistsep
2519         }%
2520         \def\@gls@noidx@loclist@finalsep{\glsnumlistlastsep}%
2521       }%
2522     }%
2523     \def\@gls@noidx@loclist@finalsep{}%
2524     \def\@gls@noidx@loclist@prev{}%
2525     \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
2526     \@gls@noidx@loclist@finalsep
2527     \@gls@noidx@loclist@prev
2528   }%
2529   {%
2530     ??\glsdoifexists{#1}%
2531     {%
2532       \GlossariesWarning{Missing location list for ‘#1’. Either
2533         a rerun is required or you haven’t referenced the entry.}%
2534     }%
2535   }%
2536 }%
2537
```

And for the number list loop:

@numberlistloop

```

2538 \newcommand*{\@glxtr@noidx@numberlistloop}[3]{%
2539   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
2540   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
2541   \let\@gls@org@glsseeformat\glsseeformat
2542   \let\glsnoidxdisplayloc#2\relax
2543   \let\glsseeformat#3\relax
2544   \ifdef\@gls@loclist
2545   {%
2546     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
2547   }%
2548   {%
2549     ??\glsdoifexists{#1}%
2550     {%
2551       \GlossariesWarning{Missing location list for ‘##1’. Either
```

```

2552         a rerun is required or you haven't referenced the entry.}%
2553     }%
2554 }%
2555 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
2556 \let\glsseeformat\@gls@org@glsseeformat
2557 }%

```

Same for entry number list.

entrynumberlist

```

2558 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
2559     \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
2560     \ifdef\@gls@loclist
2561     {%
2562         \glsnoidxloclist{\@gls@loclist}%
2563     }%
2564     {%
2565         ??\glsdoifexists{#1}%
2566         {%
2567             \GlossariesWarning{Missing location list for '#1'. Either
2568                 a rerun is required or you haven't referenced the entry.}%
2569         }%
2570     }%
2571 }%

```

entrynumberlist

```

2572 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

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

@print@glossary

```

2573 \renewcommand{\@print@glossary}{%
2574     \makeatletter
2575     \@input@{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
2576     \IfFileExists{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
2577     {}%
2578     {\glsxtrNoGlossaryWarning{\@glo@type}}%
2579     \ifglxsindy
2580         \ifcsundef{@xdy@\@glo@type @language}%
2581         {%
2582             \edef\@do@auxoutstuff{%
2583                 \noexpand\AtEndDocument{%
2584                     \noexpand\immediate\noexpand\write\@auxout{%
2585                         \string\providecommand\string\@xdylanguage[2]{}}%
2586                     \noexpand\immediate\noexpand\write\@auxout{%
2587                         \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
2588                 }%
2589             }%
2590         }%

```

```

2591 {%
2592   \edef\@do@auxoutstuff{%
2593     \noexpand\AtEndDocument{%
2594       \noexpand\immediate\noexpand\write\@auxout{%
2595         \string\providecommand\string\@xdylanguage[2]{}}%
2596       \noexpand\immediate\noexpand\write\@auxout{%
2597         \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
2598           @language\endcsname}}}%
2599     }%
2600   }%
2601 }%
2602 \@do@auxoutstuff
2603 \edef\@do@auxoutstuff{%
2604   \noexpand\AtEndDocument{%
2605     \noexpand\immediate\noexpand\write\@auxout{%
2606       \string\providecommand\string\@gls@codepage[2]{}}%
2607     \noexpand\immediate\noexpand\write\@auxout{%
2608       \string\@gls@codepage{\@glo@type}{\gls@codepage}}}%
2609   }%
2610 }%
2611 \@do@auxoutstuff
2612 \fi
2613 \renewcommand*{\@warn@nomakeglossaries}{%
2614   \GlossariesWarningNoLine{\string\makeglossaries\space
2615     hasn't been used,^^Jthe glossaries will not be updated}%
2616 }%
2617 }

```

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

`\GlsWarningHead` Header message.

```

2618 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
2619   This document is incomplete. The external file associated with
2620   the glossary '#1' (which should be called \texttt{#2})
2621   hasn't been created.%
2622 }

```

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

```

2623 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
2624   This has probably happened because there are no entries defined
2625   in this glossary.%
2626 }

```

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

```

2627 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
2628   If you don't want this glossary,
2629   add \texttt{nomain} to your package option list when you load
2630   \texttt{glossaries-extra.sty}. For example:%
2631 }

```

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

```
2632 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
2633 Did you forget to use \texttt{type=#1} when you defined your
2634 entries? If you tried to load entries into this glossary with
2635 \texttt{\string\loadglsentries} did you remember to use
2636 \texttt{[#1]} as the optional argument? If you did, check that
2637 the definitions in the file you loaded all had the type set
2638 to \texttt{\string\glsdefaulttype}.\%
2639 }
```

arningCheckFile Advisory message to check the file contents.

```
2640 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
2641 Check the contents of the file \texttt{#1}. If
2642 it's empty, that means you haven't indexed any of your entries in this
2643 glossary (using commands like \texttt{\string\gls} or
2644 \texttt{\string\glsadd}) so this list can't be generated.
2645 If the file isn't empty, the document build process hasn't been
2646 completed.\%
2647 }
```

WarningAutoMake Message when automake option has been used.

```
2648 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
2649 You may need to rerun \LaTeX. If you already have, it may be that
2650 \TeX's shell escape doesn't allow you to run
2651 \ifglxindy xindy\else makeindex\fi. Check the
2652 transcript file \texttt{\jobname.log}. If the shell escape is
2653 disabled, try one of the following:
2654
2655 \begin{itemize}
2656 \item Run the external (Lua) application:
2657
2658 \texttt{makeglossaries-lite.lua \string"\jobname\string"}
2659
2660 \item Run the external (Perl) application:
2661
2662 \texttt{makeglossaries \string"\jobname\string"}
2663 \end{itemize}
2664
2665 Then rerun \LaTeX\ on this document.
2666 \GlossariesExtraWarning{Rerun required to build the
2667 glossary '#1' or check TeX's shell escape allows
2668 you to run \ifglxindy xindy\else makeindex\fi}%
2669 }
```

WarningMisMatch Mismatching \makenoidxglossaries.

```
2670 \newcommand{\GlsXtrNoGlsWarningMisMatch}{%
2671 You need to either replace \texttt{\string\makenoidxglossaries}
2672 with \texttt{\string\makeglossaries} or replace
```

```

2673 \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
2674 \texttt{\string\printnoidxglossary}
2675 (or \texttt{\string\printnoidxglossaries}) and then rebuild
2676 this document.%
2677 }

```

arningBuildInfo Build advice.

```

2678 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
2679 Try one of the following:
2680 \begin{itemize}
2681 \item Add \texttt{automake} to your package option list when you load
2682 \texttt{glossaries-extra.sty}. For example:
2683
2684 \texttt{\string\usepackage[automake]%
2685 \glsopenbrace glossaries-extra\glsclosebrace}
2686
2687 \item Run the external (Lua) application:
2688
2689 \texttt{makeglossaries-lite.lua \string"\jobname\string"}
2690
2691 \item Run the external (Perl) application:
2692
2693 \texttt{makeglossaries \string"\jobname\string"}
2694 \end{itemize}
2695
2696 Then rerun \LaTeX\ on this document.%
2697 }

```

oGlsWarningTail Final paragraph.

```

2698 \newcommand{\GlsXtrNoGlsWarningTail}{%
2699 This message will be removed once the problem has been fixed.%
2700 }

```

GlsWarningNoOut No out file created. Build advice.

```

2701 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
2702 The file \texttt{#1} doesn't exist. This most likely means you haven't used
2703 \texttt{\string\makeglossaries} or you have used
2704 \texttt{\string\nofiles}. If this is just a draft version of the
2705 document, you can suppress this message using the
2706 \texttt{nomissingglstext} package option.%
2707 }

```

glossarywarning

```

2708 \newcommand*{@@glsxtr@defaultnoglossarywarning}[1]{%
2709 \glossarysection[\glossarytoctitle]{\glossarytitle}
2710 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
2711 \par
2712 \glsxtrifemptyglossary{#1}%
2713 {%

```

```

2714 \GlsXtrNoGlsWarningEmptyStart\space
2715 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
2716 \medskip
2717 \noindent\texttt{\string\usepackage[nomain\ifglssacronym ,acronym\fi]%
2718 \glssopenbrace glossaries-extra\glssclosbrace}
2719 \medskip
2720 }%
2721 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
2722 }%
2723 {%
2724 \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
2725 {%
2726 \GlsXtrNoGlsWarningCheckFile
2727 {\jobname.\csname @glotype@\@glo@type @out\endcsname}
2728
2729 \ifglssautomake
2730
2731 \GlsXtrNoGlsWarningAutoMake{#1}
2732
2733 \else
2734
2735 \ifthenelse{\equal{#1}{main}}%
2736 {%
2737 \GlsXtrNoGlsWarningEmptyMain\par
2738 \medskip
2739 \noindent\texttt{\string\usepackage[nomain]%
2740 \glssopenbrace glossaries-extra\glssclosbrace}
2741 \medskip
2742 }%
2743 {}%
2744
2745 \ifdefequal\makeglossaries\@no@makeglossaries
2746 {%
2747 \GlsXtrNoGlsWarningMisMatch
2748 }%
2749 {%
2750 \GlsXtrNoGlsWarningBuildInfo
2751 }%
2752 \fi
2753 }%
2754 {%
2755 \GlsXtrNoGlsWarningNoOut
2756 {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
2757 }%
2758 }%
2759 \par
2760 \GlsXtrNoGlsWarningTail
2761 }

```

Provide some commands to accompany the record option.

xtresourcefile This is provided for the benefit of any external helper application.

```

2762 \newcommand*{\glxtrresourcefile}[2][]{%
2763   \protected@write\@auxout{}{\string\glxtr@resource{#1}{#2}}%
2764   \InputIfFileExists{#2}{}%
2765   {%
2766     \GlossariesExtraWarning{No file ‘#2’}%
2767   }%
2768 }
2769 \@onlypreamble\glxtrresourcefile

```

glxtr@resource

```

2770 \newcommand*{\glxtr@resource}[2]{%

```

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

```

2771 \newcommand*{\printunsrtglossary}[1][type=\glsdefaulttype]{%
2772   \@printglossary{#1}{\@print@unsrt@glossary}%
2773 }

```

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

```

2774 \newcommand*{\printunsrtglossaries}{%
2775   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
2776 }

```

@unsrt@glossary

```

2777 \newcommand*{\@print@unsrt@glossary}{%
2778   \glossarysection[\glossarytoctitle]{\glossarytitle}%
2779   \glossarypreamble
      check for empty list
2780   \ifcseempty{glolist@\@glo@type}
2781   {%
2782     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
2783   }%
2784   {%
2785     \begin{theglossary}%
2786     \glossaryheader
2787     \glsresetentrylist
2788     \def\@gls@currentlettergroup{}%
2789     \expandafter\@for\expandafter\glscurrententrylabel\expandafter
2790       : \expandafter=\csname glolist@\@glo@type\endcsname\do{%
2791       \ifdefempty{\glscurrententrylabel}
2792       }{%
2793       {\@gls@noidx\do\glscurrententrylabel}%
2794       }%
2795     \end{theglossary}%
2796   }%
2797   \glossarypostamble
2798 }

```

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.

```
2799 \@ifpackageloaded{glossaries-accsupp}
2800 {
```

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

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

```
2801 \newcommand*{\glsaccessname}[1]{%
2802 \glsnameaccessdisplay
2803 {%
2804 \glsentryname{#1}%
2805 }%
2806 {#1}%
2807 }
```

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

```
2808 \newcommand*{\Glsaccessname}[1]{%
2809 \glsnameaccessdisplay
2810 {%
2811 \Glsentryname{#1}%
2812 }%
2813 {#1}%
2814 }
```

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

```
2815 \newcommand*{\GLSaccessname}[1]{%
2816 \glsnameaccessdisplay
2817 {%
2818 \mfirstucMakeUppercase{\glsentryname{#1}}%
2819 }%
2820 {#1}%
2821 }
```

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

```
2822 \newcommand*{\glsaccesstext}[1]{%
2823 \glstextaccessdisplay
2824 {%
2825 \glsentrytext{#1}%
2826 }%
2827 {#1}%
2828 }
```

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

```

2829 \newcommand*{\Glsaccesstext}[1]{%
2830   \glstextaccessdisplay
2831   {%
2832     \Glsentrytext{#1}%
2833   }%
2834   {#1}%
2835 }
```

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

```

2836 \newcommand*{\GLSaccesstext}[1]{%
2837   \glstextaccessdisplay
2838   {%
2839     \mfirstucMakeUppercase{\Glsentrytext{#1}}%
2840   }%
2841   {#1}%
2842 }
```

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

```

2843 \newcommand*{\glsaccessplural}[1]{%
2844   \glspluralaccessdisplay
2845   {%
2846     \Glsentryplural{#1}%
2847   }%
2848   {#1}%
2849 }
```

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

```

2850 \newcommand*{\GLSaccessplural}[1]{%
2851   \glspluralaccessdisplay
2852   {%
2853     \Glsentryplural{#1}%
2854   }%
2855   {#1}%
2856 }
```

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

```

2857 \newcommand*{\GLSaccessplural}[1]{%
2858   \glspluralaccessdisplay
2859   {%
2860     \mfirstucMakeUppercase{\Glsentryplural{#1}}%
2861   }%
2862   {#1}%
2863 }
```

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

```

2864 \newcommand*{\glsaccessfirst}[1]{%
2865   \glsfirstaccessdisplay
2866   {%
2867     \glstryfirst{#1}%
2868   }%
2869   {#1}%
2870 }

```

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

```

2871 \newcommand*{\Glsaccessfirst}[1]{%
2872   \glsfirstaccessdisplay
2873   {%
2874     \Glsstryfirst{#1}%
2875   }%
2876   {#1}%
2877 }

```

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

```

2878 \newcommand*{\GLSaccessfirst}[1]{%
2879   \glsfirstaccessdisplay
2880   {%
2881     \mfirstucMakeUppercase{\glstryfirst{#1}}%
2882   }%
2883   {#1}%
2884 }

```

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

```

2885 \newcommand*{\glsaccessfirstplural}[1]{%
2886   \glsfirstpluralaccessdisplay
2887   {%
2888     \glstryfirstplural{#1}%
2889   }%
2890   {#1}%
2891 }

```

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

```

2892 \newcommand*{\Glsaccessfirstplural}[1]{%
2893   \glsfirstpluralaccessdisplay
2894   {%
2895     \Glsstryfirstplural{#1}%
2896   }%
2897   {#1}%
2898 }

```

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

```

2899 \newcommand*{\GLSaccessfirstplural}[1]{%

```

```

2900   \glsfirstpluralaccessdisplay
2901   {%
2902       \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
2903   }%
2904   {#1}%
2905 }

```

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

```

2906   \newcommand*{\glsaccesssymbol}[1]{%
2907       \glssymbolaccessdisplay
2908       {%
2909           \glsentrysymbol{#1}%
2910       }%
2911       {#1}%
2912   }

```

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

```

2913   \newcommand*{\Glsaccesssymbol}[1]{%
2914       \glssymbolaccessdisplay
2915       {%
2916           \Glsentrysymbol{#1}%
2917       }%
2918       {#1}%
2919   }

```

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

```

2920   \newcommand*{\GLSaccesssymbol}[1]{%
2921       \glssymbolaccessdisplay
2922       {%
2923           \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
2924       }%
2925       {#1}%
2926   }

```

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

```

2927   \newcommand*{\glsaccesssymbolplural}[1]{%
2928       \glssymbolpluralaccessdisplay
2929       {%
2930           \glsentrysymbolplural{#1}%
2931       }%
2932       {#1}%
2933   }

```

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

```

2934   \newcommand*{\Glsaccesssymbolplural}[1]{%
2935       \glssymbolpluralaccessdisplay

```

```

2936     {%
2937         \Glsentrysymbolplural{#1}%
2938     }%
2939     {#1}%
2940 }

```

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

```

2941 \newcommand*{\GLSaccesssymbolplural}[1]{%
2942     \glssymbolpluralaccessdisplay
2943     {%
2944         \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
2945     }%
2946     {#1}%
2947 }

```

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

```

2948 \newcommand*{\glsaccessdesc}[1]{%
2949     \glsdescriptionaccessdisplay
2950     {%
2951         \glentrydesc{#1}%
2952     }%
2953     {#1}%
2954 }

```

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

```

2955 \newcommand*{\Glsaccessdesc}[1]{%
2956     \glsdescriptionaccessdisplay
2957     {%
2958         \Glsentrydesc{#1}%
2959     }%
2960     {#1}%
2961 }

```

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

```

2962 \newcommand*{\GLSAccessdesc}[1]{%
2963     \glsdescriptionaccessdisplay
2964     {%
2965         \mfirstucMakeUppercase{\glentrydesc{#1}}%
2966     }%
2967     {#1}%
2968 }

```

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

```

2969 \newcommand*{\glsaccessdescplural}[1]{%
2970     \glsdescriptionpluralaccessdisplay
2971     {%
2972         \glentrydescplural{#1}%

```

```

2973     }%
2974     {#1}%
2975 }

```

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

```

2976 \newcommand*\Glsaccessdescplural[1]{%
2977   \glsdescriptionpluralaccessdisplay
2978   {%
2979     \Glsentrydescplural{#1}%
2980   }%
2981   {#1}%
2982 }

```

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

```

2983 \newcommand*\GLSaccessdescplural[1]{%
2984   \glsdescriptionpluralaccessdisplay
2985   {%
2986     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
2987   }%
2988   {#1}%
2989 }

```

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

```

2990 \newcommand*\glsaccessshort[1]{%
2991   \glsshortaccessdisplay
2992   {%
2993     \glsentryshort{#1}%
2994   }%
2995   {#1}%
2996 }

```

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

```

2997 \newcommand*\Glsaccessshort[1]{%
2998   \glsshortaccessdisplay
2999   {%
3000     \Glsentryshort{#1}%
3001   }%
3002   {#1}%
3003 }

```

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

```

3004 \newcommand*\GLSAccessshort[1]{%
3005   \glsshortaccessdisplay
3006   {%
3007     \mfirstucMakeUppercase{\glsentryshort{#1}}%
3008   }%

```

```

3009     {#1}%
3010 }

```

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

```

3011 \newcommand*{\lsaccessshortpl}[1]{%
3012   \glshortpluralaccessdisplay
3013   {%
3014     \glentryshortpl{#1}%
3015   }%
3016   {#1}%
3017 }

```

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

```

3018 \newcommand*{\Lsaccessshortpl}[1]{%
3019   \glshortpluralaccessdisplay
3020   {%
3021     \Glentryshortpl{#1}%
3022   }%
3023   {#1}%
3024 }

```

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

```

3025 \newcommand*{\LSaccessshortpl}[1]{%
3026   \glshortpluralaccessdisplay
3027   {%
3028     \mfirstucMakeUppercase{\glentryshortpl{#1}}%
3029   }%
3030   {#1}%
3031 }

```

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

```

3032 \newcommand*{\glsaccesslong}[1]{%
3033   \glslongaccessdisplay{\glentrylong{#1}}{#1}%
3034 }

```

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

```

3035
3036 \newcommand*{\Glsaccesslong}[1]{%
3037   \glslongaccessdisplay{\Glentrylong{#1}}{#1}%
3038 }

```

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

```

3039 \newcommand*{\GLSaccesslong}[1]{%
3040   \glslongaccessdisplay
3041   {%
3042     \mfirstucMakeUppercase{\glentrylong{#1}}%
3043   }%

```



```

3044     {#1}%
3045 }

```

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

```

3046 \newcommand*{\glsaccesslongpl}[1]{%
3047     \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
3048 }

```

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

```

3049
3050 \newcommand*{\Glsaccesslongpl}[1]{%
3051     \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
3052 }

```

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

```

3053 \newcommand*{\GLSaccesslongpl}[1]{%
3054     \glslongpluralaccessdisplay
3055     {%
3056         \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
3057     }%
3058     {#1}%
3059 }

```

End of if part

```

3060 }
3061 {

```

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

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

```

3062 \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.

```

3063 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

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

```

3064 \newcommand*{\GLSaccessname}[1]{%
3065     \protect\mfirstucMakeUppercase{\glsentryname{#1}}}

```

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

```

3066 \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.

```

3067 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}

```

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

```

3068 \newcommand*{\GLSaccesstext}[1]{%
3069 \protect\mfirstucMakeUppercase{\glentrytext{#1}}}
```

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

```

3070 \newcommand*{\glsaccessplural}[1]{\glentryplural{#1}}
```

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

```

3071 \newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}
```

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

```

3072 \newcommand*{\GLSaccessplural}[1]{%
3073 \protect\mfirstucMakeUppercase{\glentryplural{#1}}}
```

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

```

3074 \newcommand*{\glsaccessfirst}[1]{\glentryfirst{#1}}
```

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

```

3075 \newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}
```

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

```

3076 \newcommand*{\GLSaccessfirst}[1]{%
3077 \protect\mfirstucMakeUppercase{\glentryfirst{#1}}}
```

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

```

3078 \newcommand*{\glsaccessfirstplural}[1]{\glentryfirstplural{#1}}
```

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

```

3079 \newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}
```

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

```

3080 \newcommand*{\GLSaccessfirstplural}[1]{%
3081 \protect\mfirstucMakeUppercase{\glentryfirstplural{#1}}}
```

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

```

3082 \newcommand*{\glsaccesssymbol}[1]{\glentrysymbol{#1}}
```

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

```

3083 \newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}
```

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

```

3084 \newcommand*{\GLSaccesssymbol}[1]{%
3085 \protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}
```

esssymbolplural Display the symbolplural value (no link and no check for existence).
3086 \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.
3087 \newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}

esssymbolplural Display the symbolplural value (no link and no check for existence). converted to upper case.
3088 \newcommand*{\GLSaccesssymbolplural}[1]{%
3089 \protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}

\glsaccessdesc Display the desc value (no link and no check for existence).
3090 \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.
3091 \newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}

\GLSaccessdesc Display the desc value (no link and no check for existence). converted to upper case.
3092 \newcommand*{\GLSaccessdesc}[1]{%
3093 \protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}

ccessdescplural Display the descplural value (no link and no check for existence).
3094 \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.
3095 \newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}

ccessdescplural Display the descplural value (no link and no check for existence). converted to upper case.
3096 \newcommand*{\GLSaccessdescplural}[1]{%
3097 \protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}

\glsaccessshort Display the short form (no link and no check for existence).
3098 \newcommand*{\glsaccessshort}[1]{\glsentryshort{#1}}

\Glsaccessshort Display the short form with first letter converted to uppercase (no link and no check for existence).
3099 \newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}

\GLSaccessshort Display the short value (no link and no check for existence). converted to upper case.
3100 \newcommand*{\GLSaccessshort}[1]{%
3101 \protect\mfirstucMakeUppercase{\glsentryshort{#1}}}

lsaccessshortpl Display the short plural form (no link and no check for existence).
3102 \newcommand*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}

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

```
3103 \newcommand*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

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

```
3104 \newcommand*{\GLSaccessshortpl}[1]{%
3105 \protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}
```

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

```
3106 \newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}
```

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

```
3107 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

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

```
3108 \newcommand*{\GLSaccesslong}[1]{%
3109 \protect\mfirstucMakeUppercase{\glsentrylong{#1}}}
```

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

```
3110 \newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}
```

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

```
3111 \newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

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

```
3112 \newcommand*{\GLSaccesslongpl}[1]{%
3113 \protect\mfirstucMakeUppercase{\glsentrylongpl{#1}}}
```

End of else part

```
3114 }
```

1.5 Categories

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

```
3115 \glsaddstoragekey{category}{general}{\glscategory}
```

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

```
3116 \newcommand{\glsifcategory}[4]{%
3117 \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
3118 }
```

Categories can have attributes.

categoryattribute

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

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

```
3119 \newcommand*{\glsssetcategoryattribute}[3]{%
3120   \csdef{@glssxtr@categoryattr@@#1@#2}{#3}%
3121 }
```

categoryattribute

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

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

```
3122 \newcommand*{\glssgetcategoryattribute}[2]{%
3123   \csuse{@glssxtr@categoryattr@@#1@#2}%
3124 }
```

categoryattribute

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

Tests if the category has the given attribute set.

```
3125 \newcommand*{\glsshascategoryattribute}[4]{%
3126   \ifcsvoid{@glssxtr@categoryattr@@#1@#2}{#4}{#3}%
3127 }
```

\glsssetattribute

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

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

```
3128 \newcommand*{\glsssetattribute}[3]{%
3129   \glsssetcategoryattribute{\glsscategor{#1}}{#2}{#3}%
3130 }
```

\glssgetattribute

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

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

```
3131 \newcommand*{\glssgetattribute}[2]{%
3132   \glssgetcategoryattribute{\glsscategor{#1}}{#2}%
3133 }
```

`\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.

```
3134 \newcommand*{\glshasattribute}[4]{%
3135   \ifglentryexists{#1}%
3136   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
3137   {#4}%
3138 }
```

`categoryattribute`

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

True if category has the attribute with the given value.

```
3139 \newcommand{\glcifcategoryattribute}[5]{%
3140   \ifcsundef{@glxtr@categoryattr@#1@#2}%
3141   {#5}%
3142   {\ifcsstring{@glxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
3143 }
```

`\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.

```
3144 \newcommand{\glcifattribute}[5]{%
3145   \ifglentryexists{#1}%
3146   {\glcifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
3147   {#5}%
3148 }
```

Set attributes for the default general category:

```
3149 \glsetcategoryattribute{general}{regular}{true}
```

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

```
3150 \glsetcategoryattribute{acronym}{regular}{true}
```

`regularcategory`

Convenient shortcut to create add the regular attribute.

```
3151 \newcommand*{\glsetregularcategory}[1]{%
3152   \glsetcategoryattribute{#1}{regular}{true}%
3153 }
```

regularcategory

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

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

```
3154 \newcommand{\glsifregularcategory}[3]{%  
3155   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%  
3156 }
```

notregularcategory

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

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

```
3157 \newcommand{\glsifnotregularcategory}[3]{%  
3158   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%  
3159 }
```

\glsifregular

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

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

```
3160 \newcommand{\glsifregular}[3]{%  
3161   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%  
3162 }
```

\glsifnotregular

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

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

```
3163 \newcommand{\glsifnotregular}[3]{%  
3164   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%  
3165 }
```

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.

```
3166 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
```

```

3167 \forallglossaries[#1]{#3}%
3168 {%
3169     \forallsentries[#3]{#4}%
3170     {%
3171         \glsifcategory{#4}{#2}{#5}{}%
3172     }%
3173 }%
3174 }

```

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.

```

3175 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
3176     \forallglossaries[#1]{#4}%
3177     {%
3178         \forallsentries[#4]{#5}%
3179         {%
3180             \glsifattribute{#5}{#2}{#3}{#6}{}%
3181         }%
3182     }%
3183 }

```

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

```

3184 \ifdef\newterm
3185 {%

```

\newterm

```

3186     \renewcommand*{\newterm}[2][ ]{%
3187         \newglossaryentry{#2}%
3188         {type=index,category=index,name={#2},%
3189          description={\glstrpostdescription\nopostdesc},#1}%
3190     }

```

Indexed terms are regular by default.

```

3191     \glssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

3192     \newcommand*{\glstrpostdescindex}{%
3193 }
3194 {}

```


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.

```
3195 \ifdef\printsymbols
3196 {%
```

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

```
3197 \newcommand*{\glsxtrnewsymbol}[3] [] {%
3198 \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
3199 }
```

Symbols are regular by default.

```
3200 \glsssetcategoryattribute{symbol}{regular}{true}
```

`rpostdescsymbol`

```
3201 \newcommand*{\glsxtrpostdescsymbol}{}
3202 }
3203 {}
```

Similar for the numbers option.

```
3204 \ifdef\printnumbers
3205 {%
```

`glsxtrnewnumber`

```
3206 \ifdef\printnumbers
3207 \newcommand*{\glsxtrnewnumber}[3] [] {%
3208 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
3209 }
```

Numbers are regular by default.

```
3210 \glsssetcategoryattribute{number}{regular}{true}
```

`rpostdescnumber`

```
3211 \newcommand*{\glsxtrpostdescnumber}{}
3212 }
3213 {}
```

`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.

```
3214 \newcommand*{\glsxtrsetcategory}[2] {%
3215 \@for\@glsxtr@label:=#1\do
3216 {%
3217 \glsfieldxddef{\@glsxtr@label}{category}{#2}%
3218 }%
3219 }
```

`\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.

```

3220 \newcommand*{\glxtrsetcategoryforall}[2]{%
3221   \forallglossaries[#1]{\@glxtr@type}{%
3222     \forallglsentries[\@glxtr@type]{\@glxtr@label}%
3223     {%
3224       \glsfieldxdef{\@glxtr@label}{category}{#2}%
3225     }%
3226   }%
3227 }

```

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

Apply title casing to the contents of the given field.

```

3228 \newcommand*{\glxtrfieldtitlecase}[2]{%
3229   \expandafter\glxtrfieldtitlecasesecs\expandafter
3230   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
3231 }

```

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

```

3232 \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.

```

3233 \@ifpackageloaded{glossaries-accsupp}
3234 {
3235   \renewcommand*{\glossentrydesc}[1]{%
3236     \glsdoifexistsorwarn{#1}%
3237     {%
3238       \glssetabbrvfmt{\glscategory{#1}}%

```

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

```

3239     \glshasattribute{#1}{glossdescfont}%
3240     {%
3241       \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
3242       \ifcsdef{\@glxtr@attrval}%
3243       {%
3244         \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
3245       }%
3246       {%
3247         \GlossariesExtraWarning{Unknown control sequence name
3248           '\@glxtr@attrval' supplied in glossdescfont attribute

```

```

3249         for entry '#1'. Ignoring}%
3250         \let\@glxtr@glossdescfont\@firstofone
3251     }%
3252 }%
3253 {\let\@glxtr@glossdescfont\@firstofone}%
3254 \glsifattribute{#1}{glossdesc}{firstuc}%
3255 {%
3256     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
3257 }%
3258 {%
3259     \glsifattribute{#1}{glossdesc}{title}%
3260 {%
3261     \@glxtr@do@titlecaps@warn
3262     \glsdescriptionaccessdisplay
3263     {%
3264         \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
3265     }%
3266     {#1}%
3267 }%
3268 {%
3269     \@glxtr@glossdescfont{\glsaccessdesc{#1}}%
3270 }%
3271 }%
3272 }%
3273 }
3274 }
3275 {
3276 \renewcommand*{\glossentrydesc}[1]{%
3277     \glsdoifexistsorwarn{#1}%
3278     {%
3279         \glssetabbrvfmt{\glscategory{#1}}%
3280         \glsattribute{#1}{glossdescfont}%
3281         {%
3282             \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
3283             \ifcsdef{\@glxtr@attrval}%
3284             {%
3285                 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
3286             }%
3287             {%
3288                 \GlossariesExtraWarning{Unknown control sequence name
3289                     '\@glxtr@attrval' supplied in glossdescfont attribute
3290                     for entry '#1'. Ignoring}%
3291                 \let\@glxtr@glossdescfont\@firstofone
3292             }%
3293         }%
3294         {\let\@glxtr@glossdescfont\@firstofone}%
3295         \glsifattribute{#1}{glossdesc}{firstuc}%
3296         {%
3297             \@glxtr@glossdescfont{\Glsentrydesc{#1}}%

```

```

3298     }%
3299     {%
3300         \glsifattribute{#1}{glossdesc}{title}%
3301         {%
3302             \@glsxtr@do@titlecaps@warn
3303             \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
3304         }%
3305         {%
3306             \@glsxtr@glossdescfont{\glsentrydesc{#1}}%
3307         }%
3308     }%
3309 }%
3310 }
3311 }

```

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

```

3312 \ifpackageloaded{glossaries-accsupp}
3313 {
3314   \renewcommand*{\glossentryname}[1]{%
3315     \@glsdoifexistsorwarn{#1}%
3316     {%
3317       \glssetabbrvfmt{\glscategory{#1}}%

```

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

```

3318   \glsifattribute{#1}{glossnamefont}%
3319   {%
3320     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
3321     \ifcsdef{\@glsxtr@attrval}%
3322     {%
3323       \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
3324     }%
3325     {%
3326       \GlossariesExtraWarning{Unknown control sequence name
3327         ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
3328         for entry ‘#1’. Reverting to default \string\glsnamefont}%
3329       \let\@glsxtr@glossnamefont\glsnamefont
3330     }%
3331   }%
3332   {\let\@glsxtr@glossnamefont\glsnamefont}%
3333   \glsifattribute{#1}{glossname}{firstuc}%
3334   {%
3335     \glsnameaccessdisplay
3336     {%
3337       \@glsxtr@glossnamefont{\Glsentryname{#1}}%
3338     }%
3339     {#1}%
3340   }%
3341   {%
3342     \glsifattribute{#1}{glossname}{title}%

```

```

3343      {%
3344      \@glstr@do@titlecaps@warn
3345      \glsnameaccessdisplay
3346      {%
3347      \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}}%
3348      }%
3349      {#1}%
3350      }%
3351      {%
3352      \glsifattribute{#1}{glossname}{uc}%
3353      {%
3354      \glsnameaccessdisplay
3355      {%

```

Hide the label from the upper-casing command.

```

3356      \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
3357      \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
3358      }%
3359      {#1}%
3360      }%
3361      {%
3362      \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
3363      \glsnameaccessdisplay
3364      {%
3365      \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
3366      }%
3367      {#1}%
3368      }%
3369      }%
3370      }%

```

Do post-name hook:

```

3371      \glstrpostnamehook{#1}%
3372      }%
3373  }
3374 }
3375 {
3376 \renewcommand*{\glossentryname}[1]{%
3377   \@glsdofexistsorwarn{#1}%
3378   {%
3379     \glsetabbrvfmt{\glscategory{#1}}%
3380     \glshasattribute{#1}{glossnamefont}%
3381     {%
3382       \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
3383       \ifcsdef{\@glstr@attrval}%
3384       {%
3385         \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
3386       }%
3387     }%
3388     \GlossariesExtraWarning{Unknown control sequence name

```

```

3389         ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
3390         for entry ‘#1’. Reverting to default \string\glsnamefont}%
3391         \let\@glsxtr@glossnamefont\glsnamefont
3392     }%
3393 }%
3394 {\let\@glsxtr@glossnamefont\glsnamefont}%
3395 \glsifattribute{#1}{glossname}{firstuc}%
3396 {%
3397     \@glsxtr@glossnamefont{\Glsentryname{#1}}%
3398 }%
3399 {%
3400     \glsifattribute{#1}{glossname}{title}%
3401     {%
3402         \@glsxtr@do@titlecaps@warn
3403         \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
3404     }%
3405     {%
3406         \glsifattribute{#1}{glossname}{uc}%
3407         {%

```

Hide the label from the upper-casing command.

```

3408         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
3409         \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
3410     }%
3411     {%

```

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.

```

3412         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
3413         \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
3414     }%
3415 }%
3416 }%

```

Do post-name hook.

```

3417     \glsxtrpostnamehook{#1}%
3418 }%
3419 }
3420 }

```

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

```

3421 \@ifpackageloaded{glossaries-accsupp}
3422 {
3423     \renewcommand*{\Glossentryname}[1]{%
3424         \@glsdoifexistsorwarn{#1}%
3425         {%
3426             \glssetabbrvfmt{\glscategory{#1}}%

```

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

```

3427         \glschasattribute{#1}{glossnamefont}%
3428         {%

```

```

3429     \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
3430     \ifcsdef{\@glxtr@attrval}%
3431     {%
3432         \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
3433     }%
3434     {%
3435         \GlossariesExtraWarning{Unknown control sequence name
3436             '\@glxtr@attrval' supplied in glossnamefont attribute
3437             for entry '#1'. Reverting to default \string\glsnamefont}%
3438         \let\@glxtr@glossnamefont\glsnamefont
3439     }%
3440 }%
3441 {\let\@glxtr@glossnamefont\glsnamefont}%
3442 \glsnameaccessdisplay
3443 {%
3444     \@glxtr@glossnamefont{\Glsentryname{#1}}%
3445 }%
3446 {#1}%

```

Do post-name hook:

```

3447     \glxtrpostnamehook{#1}%
3448 }%
3449 }
3450 }
3451 {
3452     \renewcommand*{\Glossentryname}[1]{%
3453         \@glsdoifexistsorwarn{#1}%
3454         {%
3455             \glsssetabbrvfmt{\glscategory{#1}}%
3456             \glshasattribute{#1}{glossnamefont}%
3457             {%
3458                 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
3459                 \ifcsdef{\@glxtr@attrval}%
3460                 {%
3461                     \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
3462                 }%
3463                 {%
3464                     \GlossariesExtraWarning{Unknown control sequence name
3465                         '\@glxtr@attrval' supplied in glossnamefont attribute
3466                         for entry '#1'. Reverting to default \string\glsnamefont}%
3467                     \let\@glxtr@glossnamefont\glsnamefont
3468                 }%
3469             }%
3470             {\let\@glxtr@glossnamefont\glsnamefont}%
3471             \@glxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

3472     \glxtrpostnamehook{#1}%
3473 }%
3474 }

```

3475 }

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.

```
3476 \newcommand*{\glstrpostnamehook}[1]{%
3477   \def\@glstrnumberformat{\glstrnumberformat}%
3478   \glstrdoautoindexname{#1}{indexname}%
```

Allow categories to hook in here.

```
3479   \csuse{\glstrpostname\glstrcategory{\glstrcurrententrylabel}}%
3480 }
```

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

```
3481 \newif\if@glstr@format@override
3482 \@glstr@format@overridefalse
```

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

`glstrFormatOverride`

```
3483 \@ifpackageloaded{hyperref}
3484 {
```

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

```
3485   \ifHy@hyperindex
3486     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
3487       \@glstr@format@override true
3488       \appto\theindex{\let\glshypernumber\@firstofone}%
3489     }
3490   \else
3491     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
3492       \@glstr@format@override true
3493       \appto\theindex{\let\glshypernumber\hyperpage}%
3494     }
3495   \fi
3496 }
3497 {
3498   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
3499     \@glstr@format@override true
3500   }
3501 }
3502 \@onlypreamble\GlsXtrEnableIndexFormatOverride
```

`glstrdoautoindexname`

```
3503 \newcommand*{\glstrdoautoindexname}[2]{%
3504   \glshasattribute{#1}{#2}%
3505   {%
```


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.

```
3506 \glstr@autoindex@setname{#1}%
```

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

```
3507 \protected@edef\glstr@attrval{\glstr@getattribute{#1}{#2}}%
3508 \ifglstr@format@override
3509 \ifdefstring{\glstr@numberformat}{\glstr@numberformat}%
3510 {\let\glstr@attrval\glstr@numberformat}%
3511 \fi
3512 \ifdefstring{\glstr@attrval}{true}%
3513 {}%
3514 {\eappto\glo@name{\glstr@autoindex@encap\glstr@attrval}}%
3515 \expandafter\index\expandafter{\glo@name}%
3516 }%
3517 {}%
3518 }
```

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

```
3519 \newcommand*{\glstr@autoindex@setname}[1]{%
3520 \def\glo@name{\string\glstryname{#1}}%
3521 \glstryfield{\glo@sort}{#1}{sort}%
3522 \gls@checkmkidxchars\glo@sort
3523 \glstr@autoindex@doextra@esc\glo@sort
3524 \epreto\glo@name{\glo@sort\glstr@autoindex@at}%
3525 }
```

dex@doextra@esc

```
3526 \newcommand*{\glstr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
3527 \ifx\glstr@autoindex@esc\gls@quotechar
3528 \else
3529 \def\gls@checkedmkidx{}%
3530 \edef\@glstr@checkspch{%
3531 \noexpand\glstr@autoindex@escquote\expandonce{#1}%
3532 \noexpand\@empty\glstr@autoindex@esc\noexpand\@nnil
3533 \glstr@autoindex@esc\noexpand\@empty\noexpand\glstr@endescspch}%
3534 \@glstr@checkspch
3535 \let#1\gls@checkedmkidx\relax
3536 \fi
```

Escape actual character unless it has already been escaped.

```
3537 \ifx\glstr@autoindex@at\gls@actualchar
3538 \else
3539 \def\gls@checkedmkidx{}%
3540 \edef\@glstr@checkspch{%
3541 \noexpand\glstr@autoindex@escat\expandonce{#1}%
3542 \noexpand\@empty\glstr@autoindex@at\noexpand\@nnil
3543 \glstr@autoindex@at\noexpand\@empty\noexpand\glstr@endescspch}%
3544 }
```

```

3544 \@@glxtr@checkspch
3545 \let#1\@gls@checkedmkidx\relax
3546 \fi

```

Escape level character unless it has already been escaped.

```

3547 \ifx\@glxtr@autoindex@level\@gls@levelchar
3548 \else
3549 \def\@gls@checkedmkidx{}%
3550 \edef\@@glxtr@checkspch{%
3551 \noexpand\@glxtr@autoindex@esclevel\expandonce{#1}%
3552 \noexpand\@empty\@glxtr@autoindex@level\noexpand\@nnil
3553 \@glxtr@autoindex@level\noexpand\@empty\noexpand\@glxtr@endescspch}%
3554 \@@glxtr@checkspch
3555 \let#1\@gls@checkedmkidx\relax
3556 \fi

```

Escape encap character unless it has already been escaped.

```

3557 \ifx\@glxtr@autoindex@encap\@gls@encapchar
3558 \else
3559 \def\@gls@checkedmkidx{}%
3560 \edef\@@glxtr@checkspch{%
3561 \noexpand\@glxtr@autoindex@escencap\expandonce{#1}%
3562 \noexpand\@empty\@glxtr@autoindex@encap\noexpand\@nnil
3563 \@glxtr@autoindex@encap\noexpand\@empty\noexpand\@glxtr@endescspch}%
3564 \@@glxtr@checkspch
3565 \let#1\@gls@checkedmkidx\relax
3566 \fi
3567 }

```

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.

```

3568 \newcommand*{\@glxtr@autoindex@at}{}

```

trSetActualChar Set the actual character.

```

3569 \newcommand*{\GlsXtrSetActualChar}[1]{%
3570 \gdef\@glxtr@autoindex@at{#1}%
3571 \def\@glxtr@autoindex@escat##1#1##2#1##3\@glxtr@endescspch{%
3572 \@@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escat}{##1}{##2}{##3}%
3573 }%
3574 }
3575 \@onlypreamble\GlsXtrSetActualChar
3576 \makeatother
3577 \GlsXtrSetActualChar{@}
3578 \makeatletter

```

autoindex@encap Encap character for use with \index.

```

3579 \newcommand*{\@glxtr@autoindex@encap}{}

```

XtrSetEncapChar Set the encap character.

```
3580 \newcommand*{\GlsXtrSetEncapChar}[1]{%
3581   \gdef\@glstr@autoindex@encap{#1}%
3582   \def\@glstr@autoindex@escencap##1##2##3\@glstr@endescspch{%
3583     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escencap}{##1}{##2}{##3}%
3584   }%
3585 }
3586 \GlsXtrSetEncapChar{||}
3587 \@onlypreamble\GlsXtrSetEncapChar
```

autoindex@level Level character for use with \index.

```
3588 \newcommand*{\@glstr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
3589 \newcommand*{\GlsXtrSetLevelChar}[1]{%
3590   \gdef\@glstr@autoindex@level{#1}%
3591   \def\@glstr@autoindex@esclevel##1##2##3\@glstr@endescspch{%
3592     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@esclevel}{##1}{##2}{##3}%
3593   }%
3594 }
3595 \GlsXtrSetLevelChar{!}
3596 \@onlypreamble\GlsXtrSetLevelChar
```

r@autoindex@esc Escape character for use with \index.

```
3597 \newcommand*{\@glstr@autoindex@esc}{"}
```

lsXtrSetEscChar Set the escape character.

```
3598 \newcommand*{\GlsXtrSetEscChar}[1]{%
3599   \gdef\@glstr@autoindex@esc{#1}%
3600   \def\@glstr@autoindex@escquote##1##2##3\@glstr@endescspch{%
3601     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escquote}{##1}{##2}{##3}%
3602   }%
3603 }
3604 \GlsXtrSetEscChar{"}
3605 \@onlypreamble\GlsXtrSetEscChar
```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```
3606 \ifdef\actualchar
3607   {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
3608   {}
```

Quote character \quotechar:

```
3609 \ifdef\quotechar
3610   {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
3611   {}
```

Level character \levelchar:

```
3612 \ifdef\levelchar
3613   {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
3614   {}
```

Encap character \encapchar:

```
3615 \ifdef\encapchar
3616 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
3617 {}}
```

leto@endescspch

```
3618 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

toindex@esc@spch

```
\@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}
```

```
3619 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
3620   \@glstmpb=\expandafter{\@glstmpb\the\@glstmpb}%
3621   \toks@={#3}%
3622   \ifx\@nnil#3\relax
3623     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
3624   \else
3625     \ifx\@nnil#4\relax
3626       \edef\@glstmpb{\the\@glstmpb\the\@glstmpb}%
3627       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
3628         #4#5\@glxtr@endescspch}%
3629     \else
3630       \edef\@glstmpb{\the\@glstmpb\the\@glstmpb}%
3631       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
3632         #2#5#1\@nnil#1\@glxtr@endescspch}%
3633     \fi
3634   \fi
3635   \@glxtr@checkspch
3636 }
```

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

```
3637 \renewcommand*{\Glossentrydesc}[1]{%
3638   \glsoifexistsorwarn{#1}%
3639   {%
3640     \glsetabbrvfmt{\glscategory{#1}}%
3641     \Glsaccessdesc{#1}%
3642   }%
3643 }
```

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

```
3644 \renewcommand*{\Glossentrysymbol}[1]{%
3645   \glsoifexistsorwarn{#1}%
3646   {%
3647     \glsetabbrvfmt{\glscategory{#1}}%
3648     \Glsaccesssymbol{#1}%
3649   }%
3650 }
```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```
3651 \renewcommand*{\Glossentrysymbol}[1]{%
3652   \glsdoifexistsorwarn{#1}%
3653   {%
3654     \glssetabbrvfmt{\glscategory{#1}}%
3655     \Glsaccesssymbol{#1}%
3656   }%
3657 }
```

Allow initials to be marked but only use the formatting for the tag in the glossary.

`\enableInitialTagging` Allow initial tagging. The first argument is a list of categories to apply this to. The second argument is the name of the command to use to tag the initials. This can't already be defined for safety unless the starred version is used.

```
3658 \newcommand*{\GlsXtrEnableInitialTagging}{%
3659   \@ifstar\s@glsextr@enabletagging\@glsextr@enabletagging
3660 }
3661 \@onlypreamble\GlsXtrEnableInitialTagging
```

`\r@enabletagging` Starred version undefines command.

```
3662 \newcommand*{\s@glsextr@enabletagging}[2]{%
3663   \undef#2%
3664   \@glsextr@enabletagging{#1}{#2}%
3665 }
```

`\r@enabletagging` Internal command.

```
3666 \newcommand*{\@glsextr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
3667   \@for\@glsextr@cat:=#1\do
3668   {%
3669     \ifdefempty\@glsextr@cat
3670     {}%
3671     {\glssetcategoryattribute{\@glsextr@cat}{tagging}{true}}%
3672   }%
3673   \newrobustcmd*#2[1]{##1}%
3674   \def\@glsextr@taggingcs{#2}%
3675   \renewcommand*\@glsextr@activate@initialtagging{%
3676     \let#2\@glsextr@tag
3677   }%
3678   \ifundef\@gls@preglossaryhook
3679   {\GlossariesExtraWarning{Initial tagging requires at least
3680     glossaries.sty v4.19 to work correctly}}%
3681   {}%
3682 }
```

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`

```
3683 \ifundef\mfu@checkword@do
3684 {
3685   \newcommand*{\mfu@checkword@do}[1]{%
3686     \ifdefstring{\mfu@checkword@arg}{#1}%
3687     {%
3688       \let\@mfu@domakefirstuc\@firstofone
3689       \listbreak
3690     }%
3691   }%
3692 }
```

`\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.

```
3693 \ifundef\mfu@checkword
3694 {
3695   \newcommand{\@glxstr@do@titlecaps@warn}{%
3696     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
3697       support not available}%

```

One warning should suffice.

```
3698   \let\@glxstr@do@titlecaps@warn\relax
3699 }
3700 }
3701 {
3702   \renewcommand*{\mfu@checkword}[1]{%
3703     \def\mfu@checkword@arg{#1}%
3704     \let\@mfu@domakefirstuc\makefirstuc
3705     \forlistloop\mfu@checkword@do\@mfu@nocaplist
3706   }
3707 }
3708 }
3709 {}% no patch required
```

`@titlecaps@warn` Do warning if title case not supported.

```
3710 \newcommand*{\@glxstr@do@titlecaps@warn}{}
```

`@initialtagging` Used in `\printglossary` but at least v4.19 of `glossaries` required.

```
3711 \newcommand*{\@glxstr@activate@initialtagging{}
```

`\@glxstr@tag` Definition of tagging command when used in glossary.

```
3712 \newrobustcmd*{\@glxstr@tag}[1]{%
3713   \glxifattribute{\glscurrententrylabel}{tagging}{true}%
3714   {\glxstrtagfont{#1}}{#1}%
3715 }
```

`\glxstrtagfont` Used in the glossary.

```
3716 \newcommand*{\glxstrtagfont}[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.

```

3717 \ifdef\@gls@preglossaryhook
3718 {
3719   \renewcommand*{\@gls@preglossaryhook}{%
3720     \@glsxtr@activate@initialtagging
3721     \let\@glsxtr@org@postdescription\glspostdescription
3722     \renewcommand*{\glspostdescription}{%
3723       \ifglsentryexists{\glscurrententrylabel}%
3724       {%
3725         \glsxtrpostdescription
3726         \@glsxtr@org@postdescription
3727       }{%
3728       }%

```

Enable the options used by `\@glsxtrp`:

```

3729   \glossxtrsetpopts
3730 }%
3731 }
3732 {}

```

`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.

```

3733 \newcommand*{\glsxtrpostdescription}{%
3734   \csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%
3735 }

```

`postdescgeneral`

```

3736 \newcommand*{\glsxtrpostdescgeneral}{}

```

`xtrpostdescterm`

```

3737 \newcommand*{\glsxtrpostdescterm}{}

```

`postdescacronym`

```

3738 \newcommand*{\glsxtrpostdescacronym}{}

```

`escabbreviation`

```

3739 \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.

```

3740 \renewcommand*{\glspostlinkhook}{%
3741 \ifglstryexists{\glslabel}{\glxtrpostlinkhook}{}%
3742 }

```

xtrpostlinkhook The entry label should already be stored in `\glslabel` by `\@gls@link`.

```

3743 \newcommand*{\glxtrpostlinkhook}{%
3744 \glxtrdiscardperiod{\glslabel}%
3745 {\glxtrpostlinkendsentence}%
3746 {\glxtrpostlink}%
3747 }

```

\glxtrpostlink

```

3748 \newcommand*{\glxtrpostlink}{%
3749 \csuse{glxtrpostlink\glscategory{\glslabel}}%
3750 }

```

linkendsentence Done by `\glxtrpostlinkhook` if a full stop is discarded.

```

3751 \newcommand*{\glxtrpostlinkendsentence}{%
3752 \ifcsdef{glxtrpostlink\glscategory{\glslabel}}
3753 {%
3754 \csuse{glxtrpostlink\glscategory{\glslabel}}%

```

Put the full stop back.

```

3755 .\spacefactor\sffcode{. \relax
3756 }%
3757 {%

```

Assume the full stop was discarded because the entry ends with a period, so adjust the space-factor.

```

3758 \spacefactor\sffcode{. \relax
3759 }%
3760 }

```

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.

```

3761 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
3762 \glxtrifwasfirstuse{\space(\glssaccessdesc{\glslabel})}{}%
3763 }

```

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.

```

3764 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
3765 \glxtrifwasfirstuse
3766 {%
3767 \ifglshassymbol{\glslabel}{\space(\glssaccesssymbol{\glslabel})}{}%
3768 }%
3769 {}%
3770 }

```


`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).

```

3771 \newcommand*{\glxtrdiscardperiod}[3]{%
3772   \glxtrifwasfirstuse
3773   {%
3774     \glusifattribute{#1}{retainfirstuseperiod}{true}%
3775     {#3}%
3776     {%
3777       \glusifattribute{#1}{discardperiod}{true}%
3778       {%
3779         \glusifplural
3780         {%
3781           \glusifattribute{#1}{pluraldiscardperiod}{true}%
3782           {\glxtrifperiod{#2}{#3}}%
3783           {#3}%
3784         }%
3785       }%
3786       \glxtrifperiod{#2}{#3}%
3787     }%
3788   }%
3789   {#3}%
3790 }%
3791 }%
3792 {%
3793   \glusifattribute{#1}{discardperiod}{true}%
3794   {%
3795     \glusifplural
3796     {%
3797       \glusifattribute{#1}{pluraldiscardperiod}{true}%
3798       {\glxtrifperiod{#2}{#3}}%
3799       {#3}%
3800     }%
3801   }%
3802   \glxtrifperiod{#2}{#3}%
3803 }%
3804 }%
3805 {#3}%
3806 }%
3807 }

```

`\glxtrifperiod` Make a convenient user command to check if the next character is a full stop (period). Works like `\@ifstar` but uses `\new@ifnextchar` rather than `\@ifnextchar`

```

3808 \newcommand*{\glxtrifperiod}[1]{\new@ifnextchar.{\@firstoftwo{#1}}}

```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`glxtr@punclist` List of characters identified as punctuation marks. (Be careful of babel shorthands!) This

doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```
3809 \newcommand*{\glxtr@punclist}{.,:;?!}
```

punctuationmark Add character to punctuation list.

```
3810 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}
```

punctuationmarks Reset the punctuation list.

```
3811 \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.)

```
3812 \newcommand*{\glxtrifnextpunc}[2]{%
3813   \def\reserved@a{#1}%
3814   \def\reserved@b{#2}%
3815   \futurelet\@glspunc@token\glxtr@ifnextpunc
3816 }
```

glxtr@ifnextpunc

```
3817 \newcommand*{\glxtr@ifnextpunc}{%
3818   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
3819   \reserved@b
3820 }
```

glxtr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
3821 \newcommand*{\glxtr@ifpunctoken}[1]{%
3822   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
3823 }
```

glxtr@ifpunctoken

```
3824 \def\@glxtr@ifpunctoken#1#2{%
3825   \let\reserved@d=#2%
3826   \ifx\reserved@d\@nnil
3827     \let\glxtr@next\@glxtr@notfoundinlist
3828   \else
3829     \ifx#1\reserved@d
3830       \let\glxtr@next\@glxtr@foundinlist
3831     \else
3832       \let\glxtr@next\@glxtr@ifpunctoken
3833     \fi
3834   \fi
3835   \glxtr@next#1%
3836 }
```

glxtr@foundinlist

```
3837 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
3838 \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.

```
3839 \newcommand{\glxstrdopostpunc}[1]{%
3840   \glxstrifnextpunc{\@glxstr@swaptwo{#1}}{#1}%
3841 }
```

@glxstr@swaptwo

```
3842 \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.

```
3843 \define@key{glxstrabbrv}{category}{%
3844   \edef\glscategorylabel{#1}%
3845   \ifcsdef{@glxstrabbrv@current@#1}%
3846   {%
```

Warning should already have been issued.

```
3847   \let\@glxstr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
3848   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
3849   \glxstr@applyabbrvstyle{\csname @glxstrabbrv@current@#1\endcsname}%
3850   \let\GlsXtrWarnDeprecatedAbbrStyle\@glxstr@orgwarndep
3851 }%
3852 {}%
3853 }
```

Save the short plural form. This may be needed before the entry is defined.

```
3854 \define@key{glxstrabbrv}{shortplural}{%
3855   \def\@glx@shortpl{#1}%
3856 }
```

Similarly for the long plural form.

```
3857 \define@key{glxstrabbrv}{longplural}{%
3858   \def\@glx@longpl{#1}%
3859 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

`\glsshortpltok`

```
3860 \newtoks\glsshortpltok
```

`\glslongpltok`

```
3861 \newtoks\glslongpltok
```

`\glsextr@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.

```
3862 \newcommand*{\@glsextr@insertdots}[2]{%
3863   \def#1{%
3864     \@glsextr@insert@dots#1#2\@nnil
3865   }}
```

`\glsextr@insert@dots`

```
3866 \newcommand*{\@glsextr@insert@dots}[2]{%
3867   \ifx\@nnil#2\relax
3868   \let\@glsextr@insert@dots@next\@gobble
3869   \else
3870   \ifx\relax#2\relax
3871   \else
3872     \appto#1{#2.}%
3873   \fi
3874   \let\@glsextr@insert@dots@next\@glsextr@insert@dots
3875   \fi
3876   \@glsextr@insert@dots@next#1%
3877 }
```

`\newabbreviation` Define a new generic abbreviation.

```
3878 \newcommand*{\newabbreviation}[4][]{%
3879   \glskeylisttok{#1}%
3880   \glslabeltok{#2}%
3881   \glsshorttok{#3}%
3882   \glslongtok{#4}%
```

Get the category.

```
3883 \def\glscategorylabel{abbreviation}%
3884 \glsextr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
3885 \setkeys*{\glsextr@abbrv}[shortplural,longplural]{#1}%
```

Set the default long plural

```
3886 \def\@gls@longpl{#4\glspluralsuffix}%
```

Has the insertdots attribute been set?

```

3887 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
3888 {%
3889   \@glxtr@insertdots\@gls@short{#3}%
3890   \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
3891   \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
3892   {%
3893     \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
3894       '\abbrvpluralsuffix}%
3895   }%
3896   {%
3897     \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
3898     {%
3899       \let\@gls@shortpl\@gls@short
3900     }%
3901     {%
3902       \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
3903         \abbrvpluralsuffix}%
3904     }%
3905   }%
3906 }%
3907 {%

```

insertdots not true.

```

3908   \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
3909   {%
3910     \def\@gls@shortpl{#3'\abbrvpluralsuffix}%
3911   }%
3912   {%
3913     \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
3914     {%
3915       \def\@gls@shortpl{#3}%
3916     }%
3917     {%
3918       \def\@gls@shortpl{#3\abbrvpluralsuffix}%
3919     }%
3920   }%
3921 }%

```

Hook for further customisation if required:

```

3922 \glxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%

```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary.

```

3923 \setkeys*{glxtrabbrv}[category]{#1}%

```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```

3924 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
3925 \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

3926 \newabbreviationhook
    Define this entry:
3927 \protected@edef\@do@newglossaryentry{%
3928     \noexpand\newglossaryentry{\the\glslabeltok}%
3929     {%
3930         type=\glstrabbrvtype,%
3931         category=abbreviation,%
3932         short={\the\glsshorttok},%
3933         shortplural={\the\glsshortpltok},%
3934         long={\the\glslongtok},%
3935         longplural={\the\glslongpltok},%
3936         name={\the\glsshorttok},%
3937         \CustomAbbreviationFields,%
3938         \the\glskeylisttok
3939     }%
3940 }%
3941 \@do@newglossaryentry
3942 \GlsXtrPostNewAbbreviation
3943 }

evpresetkeyhook Hook for extra stuff in \newabbreviation
3944 \newcommand*{\glstrnewabbspresetkeyhook}[3]{}

NewAbbreviation Hook used by abbreviation styles.
3945 \newcommand*{\GlsXtrPostNewAbbreviation}{}

bbreviationhook Hook for use with \newabbreviation.
3946 \newcommand*{\newabbreviationhook}{}

reviationFields
3947 \newcommand*{\CustomAbbreviationFields}{}

lsxtrfullformat Full format without case change.
3948 \newcommand*{\glsxtrfullformat}[2]{%
3949     \glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
3950     (\protect\glsfirstabbrvfont{\glsaccessshort{#1}})%
3951 }

lsxtrfullformat Full format with case change.
3952 \newcommand*{\Glsxtrfullformat}[2]{%
3953     \glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
3954     (\protect\glsfirstabbrvfont{\Glsaccessshort{#1}})%
3955 }

xtrfullplformat Plural full format without case change.
3956 \newcommand*{\glsxtrfullplformat}[2]{%
3957     \glsfirstlongfont{\glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
3958     (\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}})%
3959 }

```

`\glxtrfullplformat` Plural full format with case change.

```
3960 \newcommand*{\Glsxtrfullplformat}[2]{%
3961   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glxtrfullsep{#1}%
3962   (\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}})%
3963 }
```

`\glxtrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
3964 \newcommand*{\glxtrfullsep}[1]{\space}
```

In-line formats in case first use isn't compatible with `\glentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glxtrinfullformat` Full format without case change.

```
3965 \newcommand*{\glxtrinfullformat}{\glxtrfullformat}
```

`\glxtrfullformat` Full format with case change.

```
3966 \newcommand*{\Glsxtrinfullformat}{\Glsxtrfullformat}
```

`\glxtrfullplformat` Plural full format without case change.

```
3967 \newcommand*{\glxtrinfullplformat}{\glxtrfullplformat}
```

`\glxtrfullplformat` Plural full format with case change.

```
3968 \newcommand*{\Glsxtrinfullplformat}{\Glsxtrfullplformat}
```

Redefine `\glentryfull` etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the `\glxtrfull` set of commands instead.

`\glentryfull`

```
3969 \renewcommand*{\glentryfull}[1]{\glxtrinfullformat{#1}{}}
```

`\Glsentryfull`

```
3970 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinfullformat{#1}{}}
```

`\glentryfullpl`

```
3971 \renewcommand*{\glentryfullpl}[1]{\glxtrinfullplformat{#1}{}}
```

`\Glsentryfullpl`

```
3972 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinfullplformat{#1}{}}
```

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
3973 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}
```

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.

```
3974 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}
```

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.

```
3975 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}
```

bbvrdefaultfont

```
3976 \newcommand*{\glsabbrvdefaultfont}[1]{#1}
```

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.

```
3977 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}
```

longdefaultfont Default font changing command used for the long form in commands like `\glsxtrlong`.

```
3978 \newcommand*{\glslongdefaultfont}[1]{#1}
```

lsfirstlongfont Font changing command used for the long form on first use or in the full format.

```
3979 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}
```

longdefaultfont

```
3980 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}
```

brvpluralsuffix Default plural suffix.

```
3981 \newcommand*{\abbrvpluralsuffix}{\glspluralsuffix}
```

`\glsxtrfull` Full form (no case-change).

```
3982 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\@ns@glsxtrfull}
```

```
3983 \newcommand*{\ns@glsxtrfull}[2][\%]
```

```
3984 \new@ifnextchar[{\@glsxtr@full{#1}{#2}}{\%
```

```
3985 \@glsxtr@full{#1}{#2}[]}{\%
```

```
3986 }
```

`\@glsxtr@full` Low-level macro:

```
3987 \def\@glsxtr@full#1#2[#3]{\%
```

```
3988 \glsdoifexists{#2}}{\%
```

```
3989 {\%
```

```
3990 \glssetabbrvfmt{\glscategory{#2}}{\%
```

```
3991 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
3992 \let\glsifplural\@secondoftwo
```

```
3993 \let\glsupcase\@firstofthree
```

```
3994 \let\glsinsert\@empty
```

```
3995 \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.

```
3996 \glsxtrsetupfulldefs
```

```
3997 \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}}{\%
```

```
3998 }
```

```
3999 \glspostlinkhook
```

```
4000 }
```


trsetupfulldefs

```
4001 \newcommand*{\glxtrsetupfulldefs}{%
4002   \let\glxtrifwasfirstuse\@firstoftwo
4003 }
```

\Glsxtrfull Full form (first letter uppercase).

```
4004 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\@ns@Glsxtrfull}
4005 \newcommand*{\ns@Glsxtrfull}[2][]{%
4006   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}{%
4007     {\@Glsxtr@full{#1}{#2}[]}%
4008 }
```

\@Glsxtr@full Low-level macro:

```
4009 \def\@Glsxtr@full#1#2[#3]{%
4010   \glsdoifexists{#2}%
4011   {%
4012     \glssetabbrvfmt{\glscategory{#2}}%
4013     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4014     \let\glsifplural\@secondoftwo
4015     \let\glscapscase\@secondofthree
4016     \let\glsinsert\@empty
4017     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
4018     \glxtrsetupfulldefs
4019     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4020   }%
4021   \glspostlinkhook
4022 }
```

\GLSxtrfull Full form (all uppercase).

```
4023 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\@ns@GLSxtrfull}
4024 \newcommand*{\ns@GLSxtrfull}[2][]{%
4025   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}{%
4026     {\@GLSxtr@full{#1}{#2}[]}%
4027 }
```

\@GLSxtr@full Low-level macro:

```
4028 \def\@GLSxtr@full#1#2[#3]{%
4029   \glsdoifexists{#2}%
4030   {%
4031     \glssetabbrvfmt{\glscategory{#2}}%
4032     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4033     \let\glsifplural\@secondoftwo
4034     \let\glscapscase\@thirdofthree
4035     \let\glsinsert\@empty
4036     \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
4037     \glxtrsetupfulldefs
4038     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4039   }%
4040   \glspostlinkhook
```

4041 }

\glstrfullpl Plural full form (no case-change).

```
4042 \newrobustcmd*{\glstrfullpl}{\@gls@hyp@opt\ns@glstrfullpl}
4043 \newcommand*\ns@glstrfullpl[2][]{%
4044   \new@ifnextchar[{\@glstr@fullpl{#1}{#2}}%
4045                   {\@glstr@fullpl{#1}{#2}[]}%
4046 }
```

\@glstr@fullpl Low-level macro:

```
4047 \def\@glstr@fullpl#1#2[#3]{%
4048   \glsdoifexists{#2}%
4049   {%
4050     \glssetabbrvfmt{\glscategory{#2}}%
4051     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4052     \let\glsifplural\@firstoftwo
4053     \let\glscapscase\@firstofthree
4054     \let\glsinsert\@empty
4055     \def\glscustomtext{\glstrinlinefullplformat{#2}{#3}}%
4056     \glstrsetupfulldefs
4057     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4058   }%
4059   \glspostlinkhook
4060 }
```

\Glsxtrfullpl Plural full form (first letter uppercase).

```
4061 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\ns@Glsxtrfullpl}
4062 \newcommand*\ns@Glsxtrfullpl[2][]{%
4063   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
4064                   {\@Glsxtr@fullpl{#1}{#2}[]}%
4065 }
```

\@Glsxtr@fullpl Low-level macro:

```
4066 \def\@Glsxtr@fullpl#1#2[#3]{%
4067   \glsdoifexists{#2}%
4068   {%
4069     \glssetabbrvfmt{\glscategory{#2}}%
4070     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4071     \let\glsifplural\@firstoftwo
4072     \let\glscapscase\@secondofthree
4073     \let\glsinsert\@empty
4074     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
4075     \glstrsetupfulldefs
4076     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4077   }%
4078   \glspostlinkhook
4079 }
```

\GLSxtrfullpl Plural full form (all upper case).

```

4080 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\ns@GLSxtrfullpl}
4081 \newcommand*\ns@GLSxtrfullpl[2][\%
4082   \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}]{\%
4083     {\@GLSxtr@fullpl{#1}{#2}}]{\%
4084 }

```

\@GLSxtr@fullpl Low-level macro:

```

4085 \def\@GLSxtr@fullpl#1#2[#3]{\%
4086   \glsdoifexists{#2}\%
4087   {\%
4088     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4089     \let\glsifplural\@firstoftwo
4090     \let\glscapscase\@thirdofthree
4091     \let\glsinsert\@empty
4092     \def\glscustomtext{\%
4093       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}{\%
4094       \glsxtrsetupfulldefs
4095       \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}\%
4096     }\%
4097     \glspostlinkhook
4098 }

```

The short and long forms work in a similar way to acronyms.

\glsxtrshort

```

4099 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4100 \newcommand*{\ns@glsxtrshort}[2][\%
4101   \new@ifnextchar[{\@glsxtrshort{#1}{#2}}]{\@glsxtrshort{#1}{#2}}]{\%
4102 }

```

Read in the final optional argument:

```

4103 \def\@glsxtrshort#1#2[#3]{\%
4104   \glsdoifexists{#2}\%
4105   {\%

```

Need to make sure \glsabbrvfont is set correctly.

```

4106     \glssetabbrvfmt{\glscategory{#2}}{\%
4107     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4108     \let\glsxtrifwasfirstuse\@secondoftwo
4109     \let\glsifplural\@secondoftwo
4110     \let\glscapscase\@firstofthree
4111     \let\glsinsert\@empty
4112     \def\glscustomtext{\%
4113       \glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}\%
4114       \ifglsxtrininsertinside\else#3\fi
4115     }\%
4116     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}\%
4117   }\%
4118   \glspostlinkhook

```

4119 }

\Glsxtrshort

4120 \newrobustcmd*{\Glsxtrshort}{\@gls@hyp@opt\@ns@Glsxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

4121 \newcommand*{\ns@Glsxtrshort}[2] [] {%

4122 \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} [] }%

4123 }

Read in the final optional argument:

4124 \def\@Glsxtrshort#1#2[#3] {%

4125 \glsdoifexists{#2}%

4126 {%

4127 \glssetabbrvfmt{\glscategory{#2}}%

4128 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

4129 \let\glxtrifwasfirstuse\@secondoftwo

4130 \let\glsifplural\@secondoftwo

4131 \let\glscapscase\@secondofthree

4132 \let\glsinsert\@empty

4133 \def\glscustomtext{%

4134 \glsabbrvfont{\Glsaccessshort{#2}\ifglxtrininsertinside#3\fi}%

4135 \ifglxtrininsertinside\else#3\fi

4136 }%

4137 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%

4138 }%

4139 \glspostlinkhook

4140 }

\GLSxtrshort

4141 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

4142 \newcommand*{\ns@GLSxtrshort}[2] [] {%

4143 \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} [] }%

4144 }

Read in the final optional argument:

4145 \def\@GLSxtrshort#1#2[#3] {%

4146 \glsdoifexists{#2}%

4147 {%

4148 \glssetabbrvfmt{\glscategory{#2}}%

4149 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

4150 \let\glxtrifwasfirstuse\@secondoftwo

4151 \let\glsifplural\@secondoftwo

4152 \let\glscapscase\@thirdofthree

4153 \let\glsinsert\@empty

4154 \def\glscustomtext{%

4155 \mfirstucMakeUppercase

4156 {\glsabbrvfont{\Glsaccessshort{#2}\ifglxtrininsertinside#3\fi}%

4157 \ifglxtrininsertinside\else#3\fi

```

4158     }%
4159     }%
4160     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4161     }%
4162     \glspostlinkhook
4163 }

```

`\glsxtrlong`

```

4164 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
4165 \newcommand*{\ns@glsxtrlong}[2][{}]{%
4166   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2}[]}%
4167 }

    Read in the final optional argument:
4168 \def\@glsxtrlong#1#2[#3]{%
4169   \glsdoifexists{#2}%
4170   {%
4171     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4172     \let\glsxtrifwasfirstuse\@secondoftwo
4173     \let\glsifplural\@secondoftwo
4174     \let\glscapscase\@firstofthree
4175     \let\glsinsert\@empty
4176     \def\glscustomtext{%
4177       \glslongfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
4178       \ifglsxtrininsertinside\else#3\fi
4179     }%
4180     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4181     }%
4182     \glspostlinkhook
4183 }

```

`\Glsxtrlong`

```

4184 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
4185 \newcommand*{\ns@Glsxtrlong}[2][{}]{%
4186   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2}[]}%
4187 }

    Read in the final optional argument:
4188 \def\@Glsxtrlong#1#2[#3]{%
4189   \glsdoifexists{#2}%
4190   {%
4191     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4192     \let\glsxtrifwasfirstuse\@secondoftwo
4193     \let\glsifplural\@secondoftwo
4194     \let\glscapscase\@secondofthree
4195     \let\glsinsert\@empty
4196     \def\glscustomtext{%

```

```

4197      \glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
4198      \ifglxtrinsertinside\else#3\fi
4199    }%
4200    \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4201  }%
4202  \glspostlinkhook
4203 }

```

\GLSxtrlong

```

4204 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}

  Define the un-starred form. Need to determine if there is a final optional argument
4205 \newcommand*{\ns@GLSxtrlong}[2] [] {%
4206   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
4207 }

  Read in the final optional argument:
4208 \def\@GLSxtrlong#1#2[#3] {%
4209   \glsdoifexists{#2}%
4210   {%
4211     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4212     \let\glxtrifwasfirstuse\@secondoftwo
4213     \let\glsifplural\@secondoftwo
4214     \let\glscapscase\@thirdofthree
4215     \let\glsinsert\@empty
4216     \def\glscustomtext{%
4217       \mfirstucMakeUppercase
4218       {\glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
4219       \ifglxtrinsertinside\else#3\fi
4220     }%
4221   }%
4222   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4223 }%
4224 \glspostlinkhook
4225 }

```

Plural short forms:

\glxtrshortpl

```

4226 \newrobustcmd*{\glxtrshortpl}{\@gls@hyp@opt\ns@glxtrshortpl}

  Define the un-starred form. Need to determine if there is a final optional argument
4227 \newcommand*{\ns@glxtrshortpl}[2] [] {%
4228   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2} []}%
4229 }

  Read in the final optional argument:
4230 \def\@glxtrshortpl#1#2[#3] {%
4231   \glsdoifexists{#2}%
4232   {%
4233     \glssetabbrvfmt{\glscategory{#2}}%

```

```

4234 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4235 \let\glxtrifwasfirstuse\@secondoftwo
4236 \let\gl@sifplural\@firstoftwo
4237 \let\glscapscase\@firstofthree
4238 \let\gl$insert\@empty
4239 \def\glscustomtext{%
4240 \glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
4241 \ifglxtrininsertinside\else#3\fi
4242 }%
4243 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4244 }%
4245 \glspostlinkhook
4246 }

```

\Glsxtrshortpl

```

4247 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\ns@Glsxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
4248 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
4249 \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
4250 }

    Read in the final optional argument:
4251 \def\@Glsxtrshortpl#1#2[#3] {%
4252 \glsoifexists{#2}%
4253 {%
4254 \glsetabbrvfmt{\glscategory{#2}}%
4255 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4256 \let\glxtrifwasfirstuse\@secondoftwo
4257 \let\gl@sifplural\@firstoftwo
4258 \let\glscapscase\@secondofthree
4259 \let\gl$insert\@empty
4260 \def\glscustomtext{%
4261 \glabbrvfont{\Glsaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
4262 \ifglxtrininsertinside\else#3\fi
4263 }%
4264 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4265 }%
4266 \glspostlinkhook
4267 }

```

\GLSxtrshortpl

```

4268 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
4269 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
4270 \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
4271 }

    Read in the final optional argument:
4272 \def\@GLSxtrshortpl#1#2[#3] {%

```

```

4273 \glsdoifexists{#2}%
4274 {%
4275   \glssetabbrvfmt{\glscategory{#2}}%
4276   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4277   \let\glsxtrifwasfirstuse\@secondoftwo
4278   \let\glsifplural\@firstoftwo
4279   \let\glsapscase\@thirdofthree
4280   \let\glsinsert\@empty
4281   \def\glscustomtext{%
4282     \mfirstucMakeUppercase
4283     {\glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
4284     \ifglsxtrininsertinside\else#3\fi
4285   }%
4286   }%
4287   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4288 }%
4289 \glspostlinkhook
4290 }

```

Plural long forms:

\glsxtrlongpl

```

4291 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\@ns@glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4292 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
4293   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
4294 }

```

Read in the final optional argument:

```

4295 \def\@glsxtrlongpl#1#2[#3]{%
4296   \glsdoifexists{#2}%
4297   {%
4298     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4299     \let\glsxtrifwasfirstuse\@secondoftwo
4300     \let\glsifplural\@firstoftwo
4301     \let\glsapscase\@firstofthree
4302     \let\glsinsert\@empty
4303     \def\glscustomtext{%
4304       \glsfont{\glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
4305       \ifglsxtrininsertinside\else#3\fi
4306     }%
4307     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4308   }%
4309   \glspostlinkhook
4310 }

```

\Glsxtrlongpl

```

4311 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}

```


Define the un-starred form. Need to determine if there is a final optional argument

```
4312 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
4313   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
4314 }
```

Read in the final optional argument:

```
4315 \def\@Glsxtrlongpl#1#2[#3]{%
4316   \glsdoifexists{#2}%
4317   {%
4318     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4319     \let\glxtrifwasfirstuse\@secondoftwo
4320     \let\gl@ifplural\@firstoftwo
4321     \let\glscapscase\@secondofthree
4322     \let\glinsert\@empty
4323     \def\glscustomtext{%
4324       \glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
4325       \ifglxtrininsertinside\else#3\fi
4326     }%
4327     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4328   }%
4329   \glspostlinkhook
4330 }
```

\GLSxtrlongpl

```
4331 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\ns@GLSxtrlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
4332 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
4333   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
4334 }
```

Read in the final optional argument:

```
4335 \def\@GLSxtrlongpl#1#2[#3]{%
4336   \glsdoifexists{#2}%
4337   {%
4338     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4339     \let\glxtrifwasfirstuse\@secondoftwo
4340     \let\gl@ifplural\@firstoftwo
4341     \let\glscapscase\@thirdofthree
4342     \let\glinsert\@empty
4343     \def\glscustomtext{%
4344       \mfirstucMakeUppercase
4345       {\glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
4346       \ifglxtrininsertinside\else#3\fi
4347     }%
4348   }%
4349   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4350 }%
4351 \glspostlinkhook
4352 }
```

`\glssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```
4353 \newcommand*{\glssetabbrvfmt}[1]{%
4354   \ifcsdef{@glsabbrv@current@#1}%
4355     {\glsxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
4356     {\glsxtr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
4357 }
```

`sxtrgenabbrvfmt` Similar to `\glsngenacfmt`, but for abbreviations.

```
4358 \newcommand*{\glsxtrgenabbrvfmt}{%
4359   \ifdefempty\glscustomtext
4360     {%
4361       \ifglsused\glslabel
4362       {%
```

Subsequent use:

```
4363       \glsifplural
4364       {%
```

Subsequent plural form:

```
4365       \glscapscase
4366       {%
```

Subsequent plural form, don't adjust case:

```
4367       \glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert
4368       }%
4369       {%
```

Subsequent plural form, make first letter upper case:

```
4370       \glsabbrvfont{\Glsaccessshortpl{\glslabel}}\glsinsert
4371       }%
4372       {%
```

Subsequent plural form, all caps:

```
4373       \mfirstucMakeUppercase
4374       {\glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert}%
4375       }%
4376       }%
4377       {%
```

Subsequent singular form

```
4378       \glscapscase
4379       {%
```

Subsequent singular form, don't adjust case:

```
4380       \glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert
4381       }%
4382       {%
```

Subsequent singular form, make first letter upper case:

```
4383       \glsabbrvfont{\Glsaccessshort{\glslabel}}\glsinsert
4384       }%
4385       {%
```

Subsequent singular form, all caps:

```
4386      \mfirstucMakeUppercase
4387      {\glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert}%
4388      }%
4389      }%
4390      }%
4391      {%
```

First use:

```
4392      \glsifplural
4393      {%
```

First use plural form:

```
4394      \glscapscase
4395      {%
```

First use plural form, don't adjust case:

```
4396      \glstrfullplformat{\glslabel}{\glsinsert}%
4397      }%
4398      {%
```

First use plural form, make first letter upper case:

```
4399      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
4400      }%
4401      {%
```

First use plural form, all caps:

```
4402      \mfirstucMakeUppercase
4403      {\glstrfullplformat{\glslabel}{\glsinsert}}%
4404      }%
4405      }%
4406      {%
```

First use singular form

```
4407      \glscapscase
4408      {%
```

First use singular form, don't adjust case:

```
4409      \glstrfullformat{\glslabel}{\glsinsert}%
4410      }%
4411      {%
```

First use singular form, make first letter upper case:

```
4412      \Glsxtrfullformat{\glslabel}{\glsinsert}%
4413      }%
4414      {%
```

First use singular form, all caps:

```
4415      \mfirstucMakeUppercase
4416      {\glstrfullformat{\glslabel}{\glsinsert}}%
4417      }%
4418      }%
4419      }%
```

```

4420 }%
4421 {%
    User supplied text.
4422     \glscustomtext
4423 }%
4424 }

```

1.6.1 Abbreviation Styles Setup

breivationstyle

```

4425 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
4426     \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
4427     {%
4428         \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
4429     }%
4430     {%

```

Have abbreviations already been defined for this category?

```

4431     \ifcsstring{@glsabbrv@current@#1}{#2}%
4432     {%

```

Style already set.

```

4433     }%
4434     {%
4435         \def\@glxtr@dostylewarn{%
4436             \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
4437             {%
4438                 \def\@glxtr@dostylewarn{\GlossariesWarning{Abbreviation
4439                     style has been switched \MessageBreak
4440                     for category ‘#1’, \MessageBreak
4441                     but there have already been entries \MessageBreak
4442                     defined for this category. Unwanted \MessageBreak
4443                     side-effects may result}}%
4444                 \@endfortrue
4445             }%
4446             \@glxtr@dostylewarn

```

Set up the style for the given category.

```

4447         \csdef{@glsabbrv@current@#1}{#2}%
4448         \glxtr@applyabbrvstyle{#2}%
4449     }%
4450 }%
4451 }

```

applyabbrvstyle Apply the abbreviation style without existence check.

```

4452 \newcommand*{\glxtr@applyabbrvstyle}[1]{%
4453     \csuse{@glsabbrv@dispstyle@setup@#1}%
4454     \csuse{@glsabbrv@dispstyle@fmts@#1}%
4455 }

```

r@applyabbrvfmt Only apply the style formats.

```
4456 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
4457   \csuse{@glsabbrv@dispstyle@fmts@#1}%
4458 }
```

breiviationstyle 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.

```
4459 \newcommand*{\newabbreviationstyle}[3]{%
4460   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
4461     {%
4462       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
4463         defined}{}%
4464     }%
4465     {%
4466       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
4467       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
4468       #2}%
4469       \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
4470       \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
4471       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
4472       \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
4473       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
4474       #3}%
4475     }%
4476 }
```

breiviationstyle

```
4477 \newcommand*{\renewabbreviationstyle}[3]{%
4478   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
4479     {%
4480       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
4481     }%
4482     {%
4483       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
4484       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
4485       #2}%
4486       \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
4487       \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
4488       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
4489       \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
4490       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

```

4491     #3}%
4492   }%
4493 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

4494 \newcommand*{\letabbreviationstyle}[2]{%
4495   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
4496   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
4497 }

```

deprecated@abbrstyle `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

4498 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
4499   \csdef{@glsabbrv@dispstyle@setup@#1}{%
4500     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
4501     \csuse{@glsabbrv@dispstyle@setup@#2}%
4502   }%
4503   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
4504 }

```

deprecatedAbbrStyle Generate warning for deprecated style use.

```

4505 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
4506   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
4507     use ‘#2’ instead}%
4508 }

```

useAbbrStyleSetup

```

4509 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
4510   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
4511   {%
4512     \PackageError{glossaries-extra}%
4513       {Unknown abbreviation style definitions ‘#1’}{}%
4514   }%
4515   {%
4516     \csname @glsabbrv@dispstyle@setup@#1\endcsname
4517   }%
4518 }

```

useAbbrStyleFmt

```

4519 \newcommand*{\GlsXtrUseAbbrStyleFmt}[1]{%
4520   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
4521   {%
4522     \PackageError{glossaries-extra}%
4523       {Unknown abbreviation style formats ‘#1’}{}%

```

```

4524 }%
4525 {%
4526 \csname @glsabbrv@dispstyle@fnts@#1\endcsname
4527 }%
4528 }

```

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.

```

4529 \newif\ifglsxtrinsertinside
4530 \glsxtrinsertinsidefalse

```

`long-short`

```

4531 \newabbreviationstyle{long-short}%
4532 {%
4533 \renewcommand*{\CustomAbbreviationFields}{%
4534 name={\protect\glsabbrvfont{\the\glsshorttok}},
4535 sort={\the\glsshorttok},
4536 first={\protect\glsfirstlongfont{\the\glslongtok}}%
4537 \protect\glsxtrfullsep{\the\glslabeltok}}%
4538 (\protect\glsfirstabbrvfont{\the\glsshorttok}}),%
4539 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%
4540 \protect\glsxtrfullsep{\the\glslabeltok}}%
4541 (\protect\glsfirstabbrvfont{\the\glsshortpltok}}),%
4542 plural={\protect\glsabbvfont{\the\glsshortpltok}},%
4543 description={\the\glslongtok}}%

```

Unset the regular attribute if it has been set.

```

4544 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4545 \glshasattribute{\the\glslabeltok}{regular}%
4546 {%
4547 \glssetattribute{\the\glslabeltok}{regular}{false}%
4548 }%
4549 {}%
4550 }%
4551 }%
4552 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4553 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%

```

```

4554 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4555 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4556 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4557 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

4558 \renewcommand*{\glsxtrfullformat}[2]{%
4559   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
4560   \ifglsxtrininsertinside\else##2\fi
4561   \glsxtrfullsep{##1}%
4562   (\glsfirstabbrvfont{\glsaccessshort{##1}})}%
4563 }%
4564 \renewcommand*{\glsxtrfullplformat}[2]{%
4565   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
4566   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4567   (\glsfirstabbrvfont{\glsaccessshortpl{##1}})}%
4568 }%
4569 \renewcommand*{\Glsxtrfullformat}[2]{%
4570   \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
4571   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4572   (\glsfirstabbrvfont{\Glsaccessshort{##1}})}%
4573 }%
4574 \renewcommand*{\Glsxtrfullplformat}[2]{%
4575   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
4576   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4577   (\glsfirstabbrvfont{\Glsaccessshortpl{##1}})}%
4578 }%
4579 }

```

Set this as the default style for general abbreviations:

```

4580 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

4581 \newcommand*{\glsxtrlongshortdescsort}{\the\glslongtok\space(\the\glsshorttok)}

```

long-short-desc User supplies description. The long form is included in the name.

```

4582 \newabbreviationstyle{long-short-desc}%
4583 {%
4584   \renewcommand*{\CustomAbbreviationFields}{%
4585     name={\protect\glsxtrfullformat{\the\glslabeltok}{}},
4586     sort={\glsxtrlongshortdescsort},%
4587     first={\protect\glsfirstlongfont{\the\glslongtok}%
4588       \protect\glsxtrfullsep{\the\glslabeltok}%
4589       (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
4590     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
4591       \protect\glsxtrfullsep{\the\glslabeltok}%
4592       (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%

```

The text key should only have the short form.

```

4593   text={\protect\glsabbrvfont{\the\glsshorttok}},%

```



```

4594 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
4595 }%

Unset the regular attribute if it has been set.
4596 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4597   \glshasattribute{\the\glslabeltok}{regular}%
4598   {%
4599     \glissetattribute{\the\glslabeltok}{regular}{false}%
4600   }%
4601   {}%
4602 }%
4603 }%
4604 {%
4605   \GlsXtrUseAbbrStyleFmts{long-short}%
4606 }

```

short-long Short form followed by long form in parenthesis on first use.

```

4607 \newabbreviationstyle{short-long}%
4608 {%
4609   \renewcommand*{\CustomAbbreviationFields}{%
4610     name={\protect\glsabbrvfont{\the\glsshortttok}},
4611     sort={\the\glsshortttok},
4612     description={\the\glslongtok},%
4613     first={\protect\glsfirstabbrvfont{\the\glsshortttok}}%
4614     \protect\glsxtrfullsep{\the\glslabeltok}%
4615     (\protect\glsfirstlongfont{\the\glslongtok})),%
4616     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
4617     \protect\glsxtrfullsep{\the\glslabeltok}%
4618     (\protect\glsfirstlongfont{\the\glslongpltok})),%
4619     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
4620   }%
4621   \glshasattribute{\the\glslabeltok}{regular}%
4622   {%
4623     \glissetattribute{\the\glslabeltok}{regular}{false}%
4624   }%
4625   {}%
4626 }%
4627 }%
4628 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4629 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4630 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4631 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4632 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4633 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

4634 \renewcommand*{\glsxtrfullformat}[2]{%

```

```

4635 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4636 \ifglxtrinsertinside\else##2\fi
4637 \glxtrfullsep{##1}%
4638 (\glsfirstlongfont{\glsaccesslong{##1}})%
4639 }%
4640 \renewcommand*{\glxtrfullplformat}[2]{%
4641 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4642 \ifglxtrinsertinside\else##2\fi
4643 \glxtrfullsep{##1}%
4644 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4645 }%
4646 \renewcommand*{\Glsxtrfullformat}[2]{%
4647 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4648 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4649 (\glsfirstlongfont{\glsaccesslong{##1}})%
4650 }%
4651 \renewcommand*{\Glsxtrfullplformat}[2]{%
4652 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4653 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4654 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4655 }%
4656 }

```

short-long-desc User supplies description. The long form is included in the name.

```

4657 \newabbreviationstyle{short-long-desc}%
4658 {%
4659 \renewcommand*{\CustomAbbreviationFields}{%
4660 name={\protect\glxtrfullformat{\the\glslabeltok}{}},
4661 sort={\the\glsshorttok},%
4662 first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
4663 \protect\glxtrfullsep{\the\glslabeltok}%
4664 (\protect\glsfirstlongfont{\the\glslongtok}}),%
4665 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
4666 \protect\glxtrfullsep{\the\glslabeltok}%
4667 (\protect\glsfirstlongfont{\the\glslongpltok}}),%
4668 text={\protect\glsabbrvfont{\the\glsshorttok}},%
4669 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
4670 }%

```

Unset the regular attribute if it has been set.

```

4671 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4672 \glshasattribute{\the\glslabeltok}{regular}%
4673 {%
4674 \glssetattribute{\the\glslabeltok}{regular}{false}%
4675 }%
4676 {}%
4677 }%
4678 }%

```

```

4679 {%
4680   \GlsXtrUseAbbrStyleFmts{short-long}%
4681 }

```

`\longfootnotefont` Only used by the “footnote” styles.

```

4682 \newcommand*{\glfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%

```

`\longfootnotefont` Only used by the “footnote” styles.

```

4683 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%

```

`\glsextrabrvfootnote` `\glsextrabrvfootnote{<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`).

```

4684 \newcommand*{\glsextrabrvfootnote}[2]{\footnote{#2}}

```

`\footnote` Short form followed by long form in footnote on first use.

```

4685 \newabbreviationstyle{footnote}%
4686 {%
4687   \renewcommand*{\CustomAbbreviationFields}{%
4688     name={\protect\glsabbrvfont{\the\glsshorttok}},
4689     sort={\the\glsshorttok},
4690     description={\the\glslongtok},%
4691     first={\protect\glfirstabbrvfont{\the\glsshorttok}%
4692       \protect\glsextrabrvfootnote{\the\glslabeltok}%
4693       {\protect\glfirstlongfootnotefont{\the\glslongtok}}},%
4694     firstplural={\protect\glfirstabbrvfont{\the\glsshortpltok}%
4695       \protect\glsextrabrvfootnote{\the\glslabeltok}%
4696       {\protect\glfirstlongfootnotefont{\the\glslongpltok}}},%
4697     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.

```

4698   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4699     \glsssetAttribute{\the\glslabeltok}{nohyperfirst}{true}%
4700     \glshasattribute{\the\glslabeltok}{regular}%
4701     {%
4702       \glsssetAttribute{\the\glslabeltok}{regular}{false}%
4703     }%
4704   }%
4705 }%
4706 }%
4707 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4708 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4709 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4710 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4711 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
4712 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

4713 \renewcommand*{\glsxtrfullformat}[2]{%
4714   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4715   \ifglsxtrininsertinside\else##2\fi
4716   \protect\glsxtrabbrvfootnote{##1}%
4717   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
4718 }%
4719 \renewcommand*{\glsxtrfullplformat}[2]{%
4720   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4721   \ifglsxtrininsertinside\else##2\fi
4722   \protect\glsxtrabbrvfootnote{##1}%
4723   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
4724 }%
4725 \renewcommand*{\Glsxtrfullformat}[2]{%
4726   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4727   \ifglsxtrininsertinside\else##2\fi
4728   \protect\glsxtrabbrvfootnote{##1}%
4729   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
4730 }%
4731 \renewcommand*{\Glsxtrfullplformat}[2]{%
4732   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4733   \ifglsxtrininsertinside\else##2\fi
4734   \protect\glsxtrabbrvfootnote{##1}%
4735   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
4736 }%

```

The first use full form and the inline full form use the short (long) style.

```

4737 \renewcommand*{\glsxtrinlinefullformat}[2]{%
4738   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4739   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4740   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
4741 }%
4742 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
4743   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4744   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4745   (\glsfirstlongfootnotefont{\glsaccesslongpl{##1}})%
4746 }%
4747 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4748   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4749   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4750   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
4751 }%
4752 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%

```

```

4753 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4754 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4755 (\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}})%
4756 }%
4757 }

```

short-footnote

```

4758 \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.

```

4759 \newabbreviationstyle{postfootnote}%
4760 {%
4761 \renewcommand*{\CustomAbbreviationFields}{%
4762 name={\protect\glsabbrvfont{\the\glsshorttok}},
4763 sort={\the\glsshorttok},
4764 description={\the\glslongtok},%
4765 first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
4766 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
4767 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.

```

4768 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4769 \csdef{glxtrpostlink\glscategorylabel}{%
4770 \glxtrifwasfirstuse
4771 }%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

4772 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
4773 {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
4774 }%
4775 {}%
4776 }%
4777 \glshasattribute{\the\glslabeltok}{regular}%
4778 {%
4779 \glissetattribute{\the\glslabeltok}{regular}{false}%
4780 }%
4781 {}%
4782 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

4783 \renewcommand*{\glxtrsetupfulldefs}{%
4784 \let\glxtrifwasfirstuse\@secondoftwo
4785 }%
4786 }%
4787 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4788 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4789 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4790 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4791 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
4792 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

4793 \renewcommand*{\glsxtrfullformat}[2]{%
4794   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4795   \ifglsxtrininsertinside\else##2\fi
4796 }%
4797 \renewcommand*{\glsxtrfullplformat}[2]{%
4798   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4799   \ifglsxtrininsertinside\else##2\fi
4800 }%
4801 \renewcommand*{\Glsxtrfullformat}[2]{%
4802   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4803   \ifglsxtrininsertinside\else##2\fi
4804 }%
4805 \renewcommand*{\Glsxtrfullplformat}[2]{%
4806   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4807   \ifglsxtrininsertinside\else##2\fi
4808 }%

```

The first use full form and the inline full form use the short (long) style.

```

4809 \renewcommand*{\glsxtrinlinefullformat}[2]{%
4810   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4811   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4812   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
4813 }%
4814 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
4815   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4816   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4817   (\glsfirstlongfootnotefont{\glsaccesslongpl{##1}})%
4818 }%
4819 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4820   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4821   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4822   (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
4823 }%
4824 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4825   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4826   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4827   (\glsfirstlongfootnotefont{\glsaccesslongpl{##1}})%
4828 }%
4829 }

```

rt-postfootnote

```

4830 \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.

```

4831 \newabbreviationstyle{short}%
4832 {%
4833   \renewcommand*{\CustomAbbreviationFields}{%
4834     name={\protect\glsabbrvfont{\the\glsshorttok}},
4835     sort={\the\glsshorttok},
4836     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
4837     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
4838     text={\protect\glsabbrvfont{\the\glsshorttok}},
4839     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
4840     description={\the\glslongtok}}%
4841   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4842     \glssetattribute{\the\glslabeltok}{regular}{true}}%
4843 }%
4844 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4845 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4846 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4847 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4848 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4849 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

4850 \renewcommand*{\glsxtrinlinefullformat}[2]{%
4851   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
4852   \ifglsxtrininsertinside##2\fi}%
4853   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4854   (\glsfirstlongfont{\glsaccesslong{##1}})%
4855 }%
4856 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
4857   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
4858   \ifglsxtrininsertinside##2\fi}%
4859   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4860   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4861 }%
4862 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4863   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
4864   \ifglsxtrininsertinside##2\fi}%
4865   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4866   (\glsfirstlongfont{\Glsaccesslong{##1}})%
4867 }%
4868 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4869   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
4870   \ifglsxtrininsertinside##2\fi}%
4871   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4872   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%

```

4873 }%

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.

```
4874 \renewcommand*{\glxtrfullformat}[2]{%
4875   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4876   \ifglxtrinsertinside\else##2\fi
4877 }%
4878 \renewcommand*{\glxtrfullplformat}[2]{%
4879   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4880   \ifglxtrinsertinside\else##2\fi
4881 }%
4882 \renewcommand*{\Glsxtrfullformat}[2]{%
4883   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4884   \ifglxtrinsertinside\else##2\fi
4885 }%
4886 \renewcommand*{\Glsxtrfullplformat}[2]{%
4887   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4888   \ifglxtrinsertinside\else##2\fi
4889 }%
4890 }
```

Set this as the default style for acronyms:

```
4891 \setabbreviationstyle[acronym]{short}
```

short-nolong

```
4892 \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.

```
4893 \newabbreviationstyle{short-desc}%
4894 {%
4895   \renewcommand*{\CustomAbbreviationFields}{%
4896     name={\protect\glxtrinlinefullformat{\the\glslabeltok}{}},
4897     sort={\the\glsshorttok},
4898     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
4899     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
4900     text={\protect\glsabbrvfont{\the\glsshorttok}},
4901     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
4902     description={\the\glslongtok}}%
4903   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4904     \glssetattribute{\the\glslabeltok}{regular}{true}}%
4905 }%
4906 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
4907 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4908 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4909 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
```



```

4910 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4911 \renewcommand*\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

4912 \renewcommand*\glsxtrinlinefullformat}[2]{%
4913   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
4914   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4915   (\glsfirstlongfont{\glsaccesslong{##1}})%
4916 }%
4917 \renewcommand*\glsxtrinlinefullplformat}[2]{%
4918   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
4919   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4920   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4921 }%
4922 \renewcommand*\Glsxtrinlinefullformat}[2]{%
4923   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
4924   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4925   (\glsfirstlongfont{\Glsaccesslong{##1}})%
4926 }%
4927 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
4928   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
4929   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4930   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
4931 }%

```

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.

```

4932 \renewcommand*\glsxtrfullformat}[2]{%
4933   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
4934   \ifglsxtrinsertinside\else##2\fi
4935 }%
4936 \renewcommand*\glsxtrfullplformat}[2]{%
4937   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
4938   \ifglsxtrinsertinside\else##2\fi
4939 }%
4940 \renewcommand*\Glsxtrfullformat}[2]{%
4941   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
4942   \ifglsxtrinsertinside\else##2\fi
4943 }%
4944 \renewcommand*\Glsxtrfullplformat}[2]{%
4945   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
4946   \ifglsxtrinsertinside\else##2\fi
4947 }%
4948 }

```

ort-nolong-desc

```

4949 \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.

```

4950 \newabbreviationstyle{long-desc}%
4951 {%
4952   \renewcommand*{\CustomAbbreviationFields}{%
4953     name={\protect\protect\glsfirstlongfont{\the\glslongtok}},
4954     sort={\the\glslongtok},
4955     first={\protect\glsfirstlongfont{\the\glslongtok}},
4956     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
4957     text={\the\glslongtok},
4958     plural={\the\glslongpltok}%
4959   }%
4960   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4961     \glssetattribute{\the\glslabeltok}{regular}{true}}%
4962 }%
4963 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4964   \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4965   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4966   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4967   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4968   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the long format followed by the short form in parentheses.

```

4969   \renewcommand*{\glsxtrinlinefullformat}[2]{%
4970     \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
4971     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4972     (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
4973   }%
4974   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
4975     \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
4976     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4977     (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4978   }%
4979   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4980     \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
4981     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4982     (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
4983   }%
4984   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4985     \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
4986     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4987     (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4988   }%

```

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.

```

4989   \renewcommand*{\glsxtrfullformat}[2]{%
4990     \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
4991     \ifglsxtrininsertinside\else##2\fi

```

```

4992 }%
4993 \renewcommand*{\glstrfullplformat}[2]{%
4994   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
4995   \ifglstrinsertinside\else##2\fi
4996 }%
4997 \renewcommand*{\Glsxtrfullformat}[2]{%
4998   \glsfirstlongfont{\glsaccesslong{##1}\ifglstrinsertinside##2\fi}%
4999   \ifglstrinsertinside\else##2\fi
5000 }%
5001 \renewcommand*{\Glsxtrfullplformat}[2]{%
5002   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
5003   \ifglstrinsertinside\else##2\fi
5004 }%
5005 }

```

ng-noshort-desc Provide a synonym that matches similar styles.

```
5006 \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.

```

5007 \newabbreviationstyle{long}%
5008 {%
5009   \renewcommand*{\CustomAbbreviationFields}{%
5010     name={\protect\glsabbrvfont{\the\glsshorttok}},
5011     sort={\the\glsshorttok},
5012     first={\protect\glsfirstlongfont{\the\glslongtok}},
5013     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5014     text={\the\glslongtok},
5015     plural={\the\glslongpltok},%
5016     description={\the\glslongtok}%
5017   }%
5018   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5019     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5020 }%
5021 {%
5022   \GlsXtrUseAbbrStyleFmts{long-desc}%
5023 }

```

long-noshort Provide a synonym that matches similar styles.

```
5024 \letabbreviationstyle{long-noshort}{long}
```

1.6.3 Predefined Styles (Small Capitals)

These styles use:

\glstrscfont

```
5025 \newcommand*{\glstrscfont}[1]{\textsc{#1}}
```

sxtrfirstscfont

```
5026 \newcommand*{\glxtrfirstscfont}[1]{\glxtrscfont{#1}}
```

and for the default short form suffix:

\glxtrscsuffix

```
5027 \newcommand*{\glxtrscsuffix}{\glstextup{\glspluralsuffix}}
```

long-short-sc

```
5028 \newabbreviationstyle{long-short-sc}%  
5029 {%  
5030   \GlsXtrUseAbbrStyleSetup{long-short}%  
5031 }%  
5032 {%
```

Mostly as long-short style:

```
5033   \GlsXtrUseAbbrStyleFmts{long-short}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5034   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5035   \renewcommand*{\glsabbrvfont}[1]{\glxtrscfont{##1}}%  
5036   \renewcommand*{\glsfirstabbrvfont}[1]{\glxtrfirstscfont{##1}}%  
5037 }
```

g-short-sc-desc

```
5038 \newabbreviationstyle{long-short-sc-desc}%  
5039 {%  
5040   \GlsXtrUseAbbrStyleSetup{long-short-desc}%  
5041 }%  
5042 {%
```

Mostly as long-short-desc style:

```
5043   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5044   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5045   \renewcommand*{\glsabbrvfont}[1]{\glxtrscfont{##1}}%  
5046   \renewcommand*{\glsfirstabbrvfont}[1]{\glxtrfirstscfont{##1}}%  
5047 }
```

Now the short (long) version

```
5048 \newabbreviationstyle{short-sc-long}%  
5049 {%  
5050   \GlsXtrUseAbbrStyleSetup{short-long}%  
5051 }%  
5052 {%
```

Mostly as short-long style:

```
5053   \GlsXtrUseAbbrStyleFmts{short-long}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5054 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5055 \renewcommand*{\glsabbrvfont[1]{\glxtrscfont{##1}}}%
5056 \renewcommand*{\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%
5057 }
```

As before but user provides description

```
5058 \newabbreviationstyle{short-sc-long-desc}%
5059 {%
5060 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5061 }%
5062 {%
```

Mostly as short-long-desc style:

```
5063 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5064 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5065 \renewcommand*{\glsabbrvfont[1]{\glxtrscfont{##1}}}%
5066 \renewcommand*{\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%
5067 }
```

short-sc

```
5068 \newabbreviationstyle{short-sc}%
5069 {%
5070 \GlsXtrUseAbbrStyleSetup{short-nolong}%
5071 }%
5072 {%
```

Mostly as short style:

```
5073 \GlsXtrUseAbbrStyleFmts{short-nolong}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5074 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5075 \renewcommand*{\glsabbrvfont[1]{\glxtrscfont{##1}}}%
5076 \renewcommand*{\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}}%
5077 }
```

short-sc-nolong

```
5078 \letabbreviationstyle{short-sc-nolong}{short-sc}
```

short-sc-desc

```
5079 \newabbreviationstyle{short-sc-desc}%
5080 {%
5081 \GlsXtrUseAbbrStyleSetup{short-desc}%
5082 }%
5083 {%
```

Mostly as short style:

```
5084 \GlsXtrUseAbbrStyleFmts{short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5085 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5086 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5087 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
5088 }
```

-sc-nolong-desc

```
5089 \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.

```
5090 \newabbreviationstyle{long-noshort-sc}%
5091 {%
5092 \GlsXtrUseAbbrStyleSetup{long-noshort}%
5093 }%
5094 {%
```

Mostly as long style:

```
5095 \GlsXtrUseAbbrStyleFmts{long-noshort}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5096 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5097 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5098 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
5099 }
```

long-sc Backward compatibility:

```
5100 \@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.

```
5101 \newabbreviationstyle{long-noshort-sc-desc}%
5102 {%
5103 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5104 }%
5105 {%
```

Mostly as long style:

```
5106 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5107 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
5108 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
5109 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
5110 }
```

long-desc-sc Backward compatibility:

```
5111 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}
```

ort-sc-footnote

```
5112 \newabbreviationstyle{short-sc-footnote}%  
5113 {%  
5114   \GlsXtrUseAbbrStyleSetup{short-footnote}%  
5115 }%  
5116 {%
```

Mostly as long style:

```
5117   \GlsXtrUseAbbrStyleFmts{short-footnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5118   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5119   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%  
5120   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%  
5121 }
```

footnote-sc Backward compatibility:

```
5122 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}
```

sc-postfootnote

```
5123 \newabbreviationstyle{short-sc-postfootnote}%  
5124 {%  
5125   \GlsXtrUseAbbrStyleSetup{short-postfootnote}%  
5126 }%  
5127 {%
```

Mostly as long style:

```
5128   \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
5129   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
5130   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%  
5131   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%  
5132 }
```

postfootnote-sc Backward compatibility:

```
5133 \@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

```
5134 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}
```

sxtrfirstsmfont

```
5135 \newcommand*{\glxtrfirstsmfont}[1]{\glxtrsmfont{#1}}
```

and for the default short form suffix:

\glxtrmsuffix

```
5136 \newcommand*{\glxtrmsuffix}{\glspluralsuffix}
```

long-short-sm

```
5137 \newabbreviationstyle{long-short-sm}%
```

```
5138 {%
```

```
5139   \GlsXtrUseAbbrStyleSetup{long-short}%
```

```
5140 }%
```

```
5141 {%
```

Mostly as long-short style:

```
5142   \GlsXtrUseAbbrStyleFmts{long-short}%
```

```
5143   \renewcommand*{\glsabbrvfont[1]}{\glxtrsmfont{##1}}%
```

```
5144   \renewcommand*{\glsfirstabbrvfont[1]}{\glxtrfirstsmfont{##1}}%
```

```
5145   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrmsuffix}%
```

```
5146 }
```

g-short-sm-desc

```
5147 \newabbreviationstyle{long-short-sm-desc}%
```

```
5148 {%
```

```
5149   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
```

```
5150 }%
```

```
5151 {%
```

Mostly as long-short-desc style:

```
5152   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
```

```
5153   \renewcommand*{\glsabbrvfont[1]}{\glxtrsmfont{##1}}%
```

```
5154   \renewcommand*{\glsfirstabbrvfont[1]}{\glxtrfirstsmfont{##1}}%
```

```
5155   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrmsuffix}%
```

```
5156 }
```

short-sm-long Now the short (long) version

```
5157 \newabbreviationstyle{short-sm-long}%
```

```
5158 {%
```

```
5159   \GlsXtrUseAbbrStyleSetup{short-long}%
```

```
5160 }%
```

```
5161 {%
```

Mostly as short-long style:

```
5162   \GlsXtrUseAbbrStyleFmts{short-long}%
```

```
5163   \renewcommand*{\glsabbrvfont[1]}{\glxtrsmfont{##1}}%
```

```
5164   \renewcommand*{\glsfirstabbrvfont[1]}{\glxtrfirstsmfont{##1}}%
```

```
5165   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrmsuffix}%
```

```
5166 }
```

rt-sm-long-desc As before but user provides description

```
5167 \newabbreviationstyle{short-sm-long-desc}%
```

```
5168 {%
```

```
5169   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
```

```
5170 }%
```

```
5171 {%
```


Mostly as short-long-desc style:

```
5172 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
5173 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
5174 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
5175 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
5176 }
```

short-sm

```
5177 \newabbreviationstyle{short-sm}%
5178 {%
5179 \GlsXtrUseAbbrStyleSetup{short-nolong}%
5180 }%
5181 {%
```

Mostly as short style:

```
5182 \GlsXtrUseAbbrStyleFmts{short-nolong}%
5183 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
5184 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
5185 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
5186 }
```

short-sm-nolong

```
5187 \letabbreviationstyle{short-sm-nolong}{short-sm}
```

short-sm-desc

```
5188 \newabbreviationstyle{short-sm-desc}%
5189 {%
5190 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
5191 }%
5192 {%
```

Mostly as short style:

```
5193 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
5194 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
5195 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
5196 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
5197 }
```

-sm-nolong-desc

```
5198 \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`.

```
5199 \newabbreviationstyle{long-noshort-sm}%
5200 {%
5201 \GlsXtrUseAbbrStyleSetup{long-noshort}%
5202 }%
5203 {%
```

Mostly as long style:

```
5204 \GlsXtrUseAbbrStyleFmts{long-noshort}%
5205 \renewcommand*\glsabrvfont[1]{\glxtrsmfont{##1}}%
5206 \renewcommand*\glsfirstabrvfont[1]{\glxtrfirstsmfont{##1}}%
5207 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
5208 }
```

long-sm Backward compatibility:

```
5209 \@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.

```
5210 \newabbreviationstyle{long-noshort-sm-desc}%
5211 {%
5212 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5213 }%
5214 {%
```

Mostly as long style:

```
5215 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
5216 \renewcommand*\glsabrvfont[1]{\glxtrsmfont{##1}}%
5217 \renewcommand*\glsfirstabrvfont[1]{\glxtrfirstsmfont{##1}}%
5218 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
5219 }
```

long-desc-sm Backward compatibility:

```
5220 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}
```

short-sm-footnote

```
5221 \newabbreviationstyle{short-sm-footnote}%
5222 {%
5223 \GlsXtrUseAbbrStyleSetup{short-footnote}%
5224 }%
5225 {%
```

Mostly as long style:

```
5226 \GlsXtrUseAbbrStyleFmts{short-footnote}%
5227 \renewcommand*\glsabrvfont[1]{\glxtrsmfont{##1}}%
5228 \renewcommand*\glsfirstabrvfont[1]{\glxtrfirstsmfont{##1}}%
5229 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
5230 }
```

footnote-sm Backward compatibility:

```
5231 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}
```

sm-postfootnote

```
5232 \newabbreviationstyle{short-sm-postfootnote}%
5233 {%
5234 \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
```

5235 }%

5236 {%

Mostly as long style:

5237 \GlsXtrUseAbbrStyleFmts{short-postfootnote}%

5238 \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%

5239 \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}%

5240 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%

5241 }

postfootnote-sm Backward compatibility:

5242 \@glstr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

1.6.5 Predefined Styles (Emphasized)

These styles use \emph for the short form.

\glsabbrvemfont

5243 \newcommand*\glsabbrvemfont[1]{\emph{#1}}%

irstabbrvemfont

5244 \newcommand*\glsfirstabbrvemfont[1]{\glsabbrvemfont{#1}}%

firstlongemfont Only used by the “long-em” styles.

5245 \newcommand*\glsfirstlongemfont[1]{\glslongemfont{#1}}%

\glslongemfont Only used by the “long-em” styles.

5246 \newcommand*\glslongemfont[1]{\emph{#1}}%

long-short-em

5247 \newabbreviationstyle{long-short-em}%

5248 {%

5249 \GlsXtrUseAbbrStyleSetup{long-short}%

5250 }%

5251 {%

Mostly as long-short style:

5252 \GlsXtrUseAbbrStyleFmts{long-short}%

5253 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%

5254 }

g-short-em-desc

5255 \newabbreviationstyle{long-short-em-desc}%

5256 {%

5257 \GlsXtrUseAbbrStyleSetup{long-short-desc}%

5258 }%

5259 {%

Mostly as long-short-desc style:

```
5260 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
5261 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5262 }
```

ong-em-short-em

```
5263 \newabbreviationstyle{long-em-short-em}%
5264 {%
```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```
5265 \renewcommand*{\CustomAbbreviationFields}{%
5266   name={\protect\glsabbrvfont{\the\glsshorttok}},
5267   sort={\the\glsshorttok},
5268   first={\protect\glsfirstlongfont{\the\glslongtok}%
5269     \protect\glsxtrfullsep{\the\glslabeltok}%
5270     (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
5271   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
5272     \protect\glsxtrfullsep{\the\glslabeltok}%
5273     (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%
5274   plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
5275   description={\protect\glslongemfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
5276 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5277   \glshasattribute{\the\glslabeltok}{regular}%
5278   {%
5279     \glssetattribute{\the\glslabeltok}{regular}{false}%
5280   }%
5281   {}%
5282 }%
5283 }%
5284 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5285 \GlsXtrUseAbbrStyleFmts{long-short}%
5286 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5287 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5288 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5289 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5290 }
```

m-short-em-desc

```
5291 \newabbreviationstyle{long-em-short-em-desc}%
5292 {%
5293   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5294 }%
5295 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5296 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
```

```

5297 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5298 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5299 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5300 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5301 }

```

short-em-long Now the short (long) version

```

5302 \newabbreviationstyle{short-em-long}%
5303 {%
5304   \GlsXtrUseAbbrStyleSetup{short-long}%
5305 }%
5306 {%

```

Mostly as short-long style:

```

5307   \GlsXtrUseAbbrStyleFmts{short-long}%
5308   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5309   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5310 }

```

rt-em-long-desc As before but user provides description

```

5311 \newabbreviationstyle{short-em-long-desc}%
5312 {%
5313   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5314 }%
5315 {%

```

Mostly as short-long-desc style:

```

5316   \GlsXtrUseAbbrStyleFmts{short-long-desc}%
5317   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5318   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5319   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5320   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5321 }

```

hort-em-long-em

```

5322 \newabbreviationstyle{short-em-long-em}%
5323 {%

```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```

5324   \renewcommand*{\CustomAbbreviationFields}{%
5325     name={\protect\glsabbrvfont{\the\glsshorttok}},
5326     sort={\the\glsshorttok},
5327     description={\protect\glslongemfont{\the\glslongtok}},%
5328     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
5329     \protect\glsxtrfullsep{\the\glslabeltok}%
5330     (\protect\glsfirstlongfont{\the\glslongtok}}),%
5331     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
5332     \protect\glsxtrfullsep{\the\glslabeltok}%
5333     (\protect\glsfirstlongfont{\the\glslongpltok}}),%
5334     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```
5335 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5336   \glshasattribute{\the\glslabeltok}{regular}%
5337   {%
5338     \glsselattribute{\the\glslabeltok}{regular}{false}%
5339   }%
5340   {}%
5341 }%
5342 }%
5343 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5344 \GlsXtrUseAbbrStyleFmts{short-long}%
5345 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
5346 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
5347 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5348 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5349 }
```

em-long-em-desc

```
5350 \newabbreviationstyle{short-em-long-em-desc}%
5351 {%
5352   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5353 }%
5354 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5355 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
5356 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
5357 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
5358 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5359 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5360 }
```

short-em

```
5361 \newabbreviationstyle{short-em}%
5362 {%
5363   \GlsXtrUseAbbrStyleSetup{short-nolong}%
5364 }%
5365 {%
```

Mostly as short style:

```
5366 \GlsXtrUseAbbrStyleFmts{short-nolong}%
5367 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
5368 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
5369 }
```

short-em-nolong

```
5370 \letabbreviationstyle{short-em-nolong}{short-em}
```

short-em-desc

```
5371 \newabbreviationstyle{short-em-desc}%
5372 {%
5373   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
5374 }%
5375 {%
```

Mostly as short style:

```
5376   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
5377   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5378   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5379 }
```

-em-nolong-desc

```
5380 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}
```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```
5381 \newabbreviationstyle{long-noshort-em}%
5382 {%
5383   \GlsXtrUseAbbrStyleSetup{long-noshort}%
5384 }%
5385 {%
```

Mostly as long-noshort style:

```
5386   \GlsXtrUseAbbrStyleFmts{long-noshort}%
5387   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5388   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5389 }
```

long-em Backward compatibility:

```
5390 \@glsxtr@deprecated@abbrstyle{long-em}{long-noshort-em}
```

g-em-noshort-em The short form is explicitly invoked through commands like \glsshort.

```
5391 \newabbreviationstyle{long-em-noshort-em}%
5392 {%
5393   \renewcommand*\CustomAbbreviationFields{%
5394     name={\protect\glsabbrvfont{\the\glsshorttok}},
5395     sort={\the\glsshorttok},
5396     first={\protect\glsfirstlongfont{\the\glslongtok}},
5397     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5398     text={\the\glslongtok},
5399     plural={\the\glslongpltok},%
5400     description={\protect\glslongemfont{\the\glslongtok}}%
5401   }%
5402   \renewcommand*\GlsXtrPostNewAbbreviation{%
5403     \glssetattribute{\the\glslabeltok}{regular}{true}}%
5404 }%
5405 {%
```

Mostly as long-noshort style:

```
5406 \GlsXtrUseAbbrStyleFmts{long-noshort}%
5407 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5408 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5409 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
5410 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
5411 }
```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like \glsshort.

```
5412 \newabbreviationstyle{long-noshort-em-desc}%
5413 {%
5414 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5415 }%
5416 {%
```

Mostly as long style:

```
5417 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
5418 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5419 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5420 }
```

long-desc-em Backward compatibility:

```
5421 \@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.

```
5422 \newabbreviationstyle{long-em-noshort-em-desc}%
5423 {%
5424 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5425 }%
5426 {%
```

Mostly as long style:

```
5427 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
5428 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5429 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5430 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
5431 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
5432 }
```

ort-em-footnote

```
5433 \newabbreviationstyle{short-em-footnote}%
5434 {%
5435 \GlsXtrUseAbbrStyleSetup{short-footnote}%
5436 }%
5437 {%
```


Mostly as long style:

```
5438 \GlsXtrUseAbbrStyleFmts{short-footnote}%
5439 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5440 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5441 }
```

footnote-em Backward compatibility:

```
5442 \@glsxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}
```

em-postfootnote

```
5443 \newabbreviationstyle{short-em-postfootnote}%
5444 {%
5445 \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
5446 }%
5447 {%
```

Mostly as long style:

```
5448 \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
5449 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5450 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5451 }
```

postfootnote-em Backward compatibility:

```
5452 \@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.

```
5453 \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.

```
5454 \ifdef\glscurrentfieldvalue
5455 {
5456 \newcommand*\glsxtruserparen[2]{%
5457 \glsxtrfullsep{#2}%
5458 (#1\ifglshasfield{\glsxtruserfield}{#2}{, \glscurrentfieldvalue}{})%
5459 }
5460 }
5461 {
5462 \newcommand*\glsxtruserparen[2]{%
5463 \glsxtrfullsep{#2}%
5464 (#1\ifglshasfield{\glsxtruserfield}{#2}{, \@glo@thisvalue}{})%
5465 }
5466 }
```

Font used for short form:

lsabbrvuserfont

```
5467 \newcommand*{\glsabbrvuserfont}[1]{#1}
```

Font used for short form on first use:

stabbrvuserfont

```
5468 \newcommand*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}
```

Font used for long form:

glslonguserfont

```
5469 \newcommand*{\glslonguserfont}[1]{#1}
```

Font used for long form on first use:

rstlonguserfont

```
5470 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
5471 \newcommand*{\glsxtrusersuffix}{\glspluralsuffix}
```

long-short-user

```
5472 \newabbreviationstyle{long-short-user}%
```

```
5473 {%
```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```
5474 \renewcommand*{\CustomAbbreviationFields}{%
```

```
5475   name={\protect\glsabbrvfont{\the\glsshorttok}},
```

```
5476   sort={\the\glsshorttok},
```

```
5477   first={\protect\glsfirstlongfont{\the\glslongtok}}%
```

```
5478   \protect\glsxtruserparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}{\the\glslabeltok}},
```

```
5479   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%
```

```
5480   \protect\glsxtruserparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}{\the\glslabeltok}},
```

```
5481   plural={\protect\glsabbbfont{\the\glsshortpltok}},%
```

```
5482   description={\protect\glslonguserfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
5483 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
5484   \glshasattribute{\the\glslabeltok}{regular}}%
```

```
5485   {%
```

```
5486     \glssetattribute{\the\glslabeltok}{regular}{false}}%
```

```
5487   }%
```

```
5488   {}%
```

```
5489 }%
```

```
5490 }%
```

```
5491 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```

5492 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
5493 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
5494 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
5495 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
5496 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5497 \renewcommand*{\glxtrfullformat}[2]{%
5498   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
5499   \ifglxtrininsertinside\else##2\fi
5500   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
5501 }%
5502 \renewcommand*{\glxtrfullplformat}[2]{%
5503   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
5504   \ifglxtrininsertinside\else##2\fi
5505   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
5506 }%
5507 \renewcommand*{\Glsxtrfullformat}[2]{%
5508   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
5509   \ifglxtrininsertinside\else##2\fi
5510   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}{##1}%
5511 }%
5512 \renewcommand*{\Glsxtrfullplformat}[2]{%
5513   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
5514   \ifglxtrininsertinside\else##2\fi
5515   \glxtruserparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}{##1}%
5516 }%
5517 }

```

short-user-desc

```

5518 \newabbreviationstyle{long-short-user-desc}%
5519 {%
5520   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5521 }%
5522 {%
5523   \GlsXtrUseAbbrStyleFmts{long-short-user}%
5524 }

```

short-long-user

```

5525 \newabbreviationstyle{short-long-user}%
5526 {%

```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```

5527 \renewcommand*{\CustomAbbreviationFields}{%
5528   name={\protect\glsabbrvfont{\the\glsshorttok}},
5529   sort={\the\glsshorttok},
5530   description={\protect\glslonguserfont{\the\glslongtok}},%
5531   first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
5532   \protect\glxtruserparen{\protect\glslonguserfont{\the\glslongtok}}{\the\glslabeltok}},%

```

```

5533   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
5534   \protect\glsxtruserparen{\protect\glslonguserfont{\the\glslongpltok}}{\the\glslabeltok}},%
5535   plural={\protect\glsabbvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

5536   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5537   \glshasattribute{\the\glslabeltok}{regular}%
5538   {%
5539   \glssetattribute{\the\glslabeltok}{regular}{false}%
5540   }%
5541   {}}%
5542   }%
5543 }%
5544 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5545   \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
5546   \renewcommand*{\glsabbrvfont[1]}{\glsabbrvuserfont{##1}}%
5547   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
5548   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
5549   \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5550   \renewcommand*{\glsxtrfullformat}[2]{%
5551   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5552   \ifglxtrinsertinside\else##2\fi
5553   \glsxtruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%
5554   }%
5555   \renewcommand*{\glsxtrfullplformat}[2]{%
5556   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5557   \ifglxtrinsertinside\else##2\fi
5558   \glsxtruserparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}{##1}%
5559   }%
5560   \renewcommand*{\Glsxtrfullformat}[2]{%
5561   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
5562   \ifglxtrinsertinside\else##2\fi
5563   \glsxtruserparen{\glsfirstlongfont{\Glsaccesslong{##1}}}{##1}%
5564   }%
5565   \renewcommand*{\Glsxtrfullplformat}[2]{%
5566   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
5567   \ifglxtrinsertinside\else##2\fi
5568   \glsxtruserparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}{##1}%
5569   }%
5570 }

```

-long-user-desc

```

5571 \newabbreviationstyle{short-long-user-desc}%
5572 {%
5573   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5574 }%

```

```

5575 {%
5576   \GlsXtrUseAbbrStyleFmts{short-long-user}%
5577 }

```

1.7 Using Entries in Headings

There are four main problems with using entries in sectioning commands: they can mess with the first use flag if they end up in the table of contents, they can add unwanted numbers to the entry's location list, the label is corrupted if used inside `\MakeUppercase` (which is used by the default headings style) and they need to be expandable for PDF bookmarks. The glossaries package therefore recommends the use of the expandable commands, such as `\glsentryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glsabbrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref`'s `\texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glsxtrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to `false`, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that `glossaries` automatically loads `textcase`, so the label can be protected from case change with `textcase`'s `\NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
5578 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```

5579 \renewcommand*{\markright}[1]{%
5580   \glsxtrmarkhook
5581   \@glsxtr@org@markright{\@glsxtrinmark#1\@glsxtrnotinmark}%
5582   \glsxtrrestoremarkhook
5583 }

```

`\markboth` Save original definition:

```
5584 \let\@glsxtr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```

5585 \renewcommand*{\markboth}[2]{%
5586   \glsxtrmarkhook
5587   \@glsxtr@org@markboth
5588   {\@glsxtrinmark#1\@glsxtrnotinmark}%

```

```

5589   {\@glsxtrinmark#2\@glsxtrnotinmark}%
5590   \glsxtrrestoremarkhook
5591 }

```

If this causes a problem provide a simple way of switching back to the original definitions:

`\glsxtrRevertMarks`

```

5592 \newcommand*{\glsxtrRevertMarks}{%
5593   \let\markright\@glsxtr@org@markright
5594   \let\markboth\@glsxtr@org@markboth
5595 }

```

`\glsxtrifinmark`

```

5596 \newcommand*{\glsxtrifinmark}[2]{#2}

```

`\@glsxtrinmark`

```

5597 \newrobustcmd*{\@glsxtrinmark}{%
5598   \let\glsxtrifinmark\@firstoftwo
5599 }

```

`\glsxtrnotinmark`

```

5600 \newrobustcmd*{\@glsxtrnotinmark}{%
5601   \let\glsxtrifinmark\@secondoftwo
5602 }

```

`\glsxtrmarkhook` Hook used in new definition of `\markboth` and `\markright` to make some changes to apply to the marks:

```

5603 \newcommand*{\glsxtrmarkhook}{%

```

Save current definitions:

```

5604   \let\@glsxtr@org@MakeUppercase\MakeUppercase
5605   \let\@glsxtr@org@glsxtrtitleshort\glsxtrtitleshort
5606   \let\@glsxtr@org@glsxtrtitleshortpl\glsxtrtitleshortpl
5607   \let\@glsxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
5608   \let\@glsxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
5609   \let\@glsxtr@org@glsxtrtitletext\glsxtrtitletext
5610   \let\@glsxtr@org@Glsxtrtitletext\Glsxtrtitletext
5611   \let\@glsxtr@org@glsxtrtitleplural\glsxtrtitleplural
5612   \let\@glsxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
5613   \let\@glsxtr@org@glsxtrtitlefirst\glsxtrtitlefirst
5614   \let\@glsxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
5615   \let\@glsxtr@org@glsxtrtitlefirstplural\glsxtrtitlefirstplural
5616   \let\@glsxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
5617   \let\@glsxtr@org@glsxtrtitlelong\glsxtrtitlelong
5618   \let\@glsxtr@org@glsxtrtitlelongpl\glsxtrtitlelongpl
5619   \let\@glsxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
5620   \let\@glsxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
5621   \let\@glsxtr@org@glsxtrtitlefull\glsxtrtitlefull
5622   \let\@glsxtr@org@glsxtrtitlefullpl\glsxtrtitlefullpl

```

```

5623 \let\@glsxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
5624 \let\@glsxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl

```

New definitions

```

5625 \let\glsxtrifinmark\@firstoftwo
5626 \let\MakeUppercase\MakeTextUppercase
5627 \let\glsxtrtitleshort\glsxtrheadshort
5628 \let\glsxtrtitleshortpl\glsxtrheadshortpl
5629 \let\Glsxtrtitleshort\Glsxtrheadshort
5630 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
5631 \let\glsxtrtitletext\glsxtrheadtext
5632 \let\Glsxtrtitletext\Glsxtrheadtext
5633 \let\glsxtrtitleplural\glsxtrheadplural
5634 \let\Glsxtrtitleplural\Glsxtrheadplural
5635 \let\glsxtrtitlefirst\glsxtrheadfirst
5636 \let\Glsxtrtitlefirst\Glsxtrheadfirst
5637 \let\glsxtrtitlefirstplural\glsxtrheadfirstplural
5638 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
5639 \let\glsxtrtitlelong\glsxtrheadlong
5640 \let\glsxtrtitlelongpl\glsxtrheadlongpl
5641 \let\Glsxtrtitlelong\Glsxtrheadlong
5642 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
5643 \let\glsxtrtitlefull\glsxtrheadfull
5644 \let\glsxtrtitlefullpl\glsxtrheadfullpl
5645 \let\Glsxtrtitlefull\Glsxtrheadfull
5646 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
5647 }

```

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.)

```

5648 \newcommand*{\glsxtrrestoremarkhook}{%
5649 \let\glsxtrifinmark\@secondoftwo
5650 \let\MakeUppercase\@glsxtr@org@MakeUppercase
5651 \let\glsxtrtitleshort\@glsxtr@org@glsxtrtitleshort
5652 \let\glsxtrtitleshortpl\@glsxtr@org@glsxtrtitleshortpl
5653 \let\Glsxtrtitleshort\@glsxtr@org@Glsxtrtitleshort
5654 \let\Glsxtrtitleshortpl\@glsxtr@org@Glsxtrtitleshortpl
5655 \let\glsxtrtitletext\@glsxtr@org@glsxtrtitletext
5656 \let\Glsxtrtitletext\@glsxtr@org@Glsxtrtitletext
5657 \let\glsxtrtitleplural\@glsxtr@org@glsxtrtitleplural
5658 \let\Glsxtrtitleplural\@glsxtr@org@Glsxtrtitleplural
5659 \let\glsxtrtitlefirst\@glsxtr@org@glsxtrtitlefirst
5660 \let\Glsxtrtitlefirst\@glsxtr@org@Glsxtrtitlefirst
5661 \let\glsxtrtitlefirstplural\@glsxtr@org@glsxtrtitlefirstplural
5662 \let\Glsxtrtitlefirstplural\@glsxtr@org@Glsxtrtitlefirstplural
5663 \let\glsxtrtitlelong\@glsxtr@org@glsxtrtitlelong
5664 \let\glsxtrtitlelongpl\@glsxtr@org@glsxtrtitlelongpl

```

```

5665 \let\Glsxtrtitlelong\@glsxtr@org@Glsxtrtitlelong
5666 \let\Glsxtrtitlelongpl\@glsxtr@org@Glsxtrtitlelongpl
5667 \let\glsxtrtitlefull\@glsxtr@org@glsxtrtitlefull
5668 \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
5669 \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
5670 \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl
5671 }

```

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.

```

5672 \newcommand*{\glsxtrheadshort}[1]{%
5673   \protect\NoCaseChange
5674   {%
5675     \glsifattribute{#1}{headuc}{true}%
5676     {%
5677       \GLSxtrshort[noindex,hyper=false]{#1}[]%
5678     }%
5679     {%
5680       \glsxtrshort[noindex,hyper=false]{#1}[]%
5681     }%
5682   }%
5683 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

5684 \newrobustcmd*{\lsxtrtitleshort}[1]{%
5685   \glsxtrshort[noindex,hyper=false]{#1}[]%
5686 }

```

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.

```

5687 \newcommand*{\glsxtrheadshortpl}[1]{%
5688   \protect\NoCaseChange
5689   {%
5690     \glsifattribute{#1}{headuc}{true}%
5691     {%
5692       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
5693     }%
5694     {%
5695       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
5696     }%
5697   }%
5698 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```

5699 \newrobustcmd*{\lsxtrtitleshortpl}[1]{%

```



```

5700 \glsxtrshortpl[noindex,hyper=false]{#1}[]%
5701 }

```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```

5702 \newcommand*{\Glsxtrheadshort}[1]{%
5703 \protect\NoCaseChange
5704 {%
5705 \glsifattribute{#1}{headuc}{true}%
5706 {%
5707 \Glsxtrshort[noindex,hyper=false]{#1}[]%
5708 }%
5709 {%
5710 \Glsxtrshort[noindex,hyper=false]{#1}[]%
5711 }%
5712 }%
5713 }

```

ltxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5714 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
5715 \Glsxtrshort[noindex,hyper=false]{#1}[]%
5716 }

```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```

5717 \newcommand*{\Glsxtrheadshortpl}[1]{%
5718 \protect\NoCaseChange
5719 {%
5720 \glsifattribute{#1}{headuc}{true}%
5721 {%
5722 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
5723 }%
5724 {%
5725 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
5726 }%
5727 }%
5728 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5729 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
5730 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
5731 }

```

\glsxtrheadtext As above but for the text value.

```

5732 \newcommand*{\glsxtrheadtext}[1]{%
5733 \protect\NoCaseChange

```

```

5734 {%
5735   \glsifattribute{#1}{headuc}{true}%
5736   {%
5737     \GLStext[noindex,hyper=false]{#1}[]%
5738   }%
5739   {%
5740     \glstext[noindex,hyper=false]{#1}[]%
5741   }%
5742 }%
5743 }

```

`\glsxtrtitletext` Command to display text value in section title and table of contents.

```

5744 \newrobustcmd*{\glsxtrtitletext}[1]{%
5745   \glstext[noindex,hyper=false]{#1}[]%
5746 }

```

`\Glsxtrheadtext` First letter converted to upper case

```

5747 \newcommand*{\Glsxtrheadtext}[1]{%
5748   \protect\NoCaseChange
5749   {%
5750     \glsifattribute{#1}{headuc}{true}%
5751     {%
5752       \GLStext[noindex,hyper=false]{#1}[]%
5753     }%
5754     {%
5755       \Glstext[noindex,hyper=false]{#1}[]%
5756     }%
5757   }%
5758 }

```

`\Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```

5759 \newrobustcmd*{\Glsxtrtitletext}[1]{%
5760   \Glstext[noindex,hyper=false]{#1}[]%
5761 }

```

`\lsxtrheadplural` As above but for the plural value.

```

5762 \newcommand*{\lsxtrheadplural}[1]{%
5763   \protect\NoCaseChange
5764   {%
5765     \glsifattribute{#1}{headuc}{true}%
5766     {%
5767       \GLSplural[noindex,hyper=false]{#1}[]%
5768     }%
5769     {%
5770       \glsplural[noindex,hyper=false]{#1}[]%
5771     }%
5772   }%
5773 }

```

sxtrtitleplural Command to display plural value in section title and table of contents.

```
5774 \newrobustcmd*{\glsxtrtitleplural}[1]{%
5775   \glsplural[noindex,hyper=false]{#1}[]%
5776 }
```

lsxtrheadplural Convert first letter to upper case.

```
5777 \newcommand*{\Glsxtrheadplural}[1]{%
5778   \protect\NoCaseChange
5779   {%
5780     \glusifattribute{#1}{headuc}{true}%
5781     {%
5782       \GLSplural[noindex,hyper=false]{#1}[]%
5783     }%
5784     {%
5785       \Glsplural[noindex,hyper=false]{#1}[]%
5786     }%
5787   }%
5788 }
```

sxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
5789 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
5790   \Glsplural[noindex,hyper=false]{#1}[]%
5791 }
```

glxtrheadfirst As above but for the first value.

```
5792 \newcommand*{\glxtrheadfirst}[1]{%
5793   \protect\NoCaseChange
5794   {%
5795     \glusifattribute{#1}{headuc}{true}%
5796     {%
5797       \GLSfirst[noindex,hyper=false]{#1}[]%
5798     }%
5799     {%
5800       \glsfirst[noindex,hyper=false]{#1}[]%
5801     }%
5802   }%
5803 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```
5804 \newrobustcmd*{\glxtrtitlefirst}[1]{%
5805   \glsfirst[noindex,hyper=false]{#1}[]%
5806 }
```

Glsxtrheadfirst First letter converted to upper case

```
5807 \newcommand*{\Glsxtrheadfirst}[1]{%
5808   \protect\NoCaseChange
5809   {%
```

```

5810 \glsifattribute{#1}{headuc}{true}%
5811 {%
5812 \GLSfirst[noindex,hyper=false]{#1}[]%
5813 }%
5814 {%
5815 \GLSfirst[noindex,hyper=false]{#1}[]%
5816 }%
5817 }%
5818 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```

5819 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
5820 \GLSfirst[noindex,hyper=false]{#1}[]%
5821 }

```

headfirstplural As above but for the firstplural value.

```

5822 \newcommand*{\Glsxtrheadfirstplural}[1]{%
5823 \protect\NoCaseChange
5824 {%
5825 \glsifattribute{#1}{headuc}{true}%
5826 {%
5827 \GLSfirstplural[noindex,hyper=false]{#1}[]%
5828 }%
5829 {%
5830 \Glsfirstplural[noindex,hyper=false]{#1}[]%
5831 }%
5832 }%
5833 }

```

itlefirstplural Command to display firstplural value in section title and table of contents.

```

5834 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
5835 \Glsfirstplural[noindex,hyper=false]{#1}[]%
5836 }

```

headfirstplural First letter converted to upper case

```

5837 \newcommand*{\Glsxtrheadfirstplural}[1]{%
5838 \protect\NoCaseChange
5839 {%
5840 \glsifattribute{#1}{headuc}{true}%
5841 {%
5842 \GLSfirstplural[noindex,hyper=false]{#1}[]%
5843 }%
5844 {%
5845 \Glsfirstplural[noindex,hyper=false]{#1}[]%
5846 }%
5847 }%
5848 }

```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
5849 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
5850   \Glsfirstplural[noindex,hyper=false]{#1}[]%
5851 }
```

`\glxtrheadlong` Command used to display long form in the page header.

```
5852 \newcommand*{\glxtrheadlong}[1]{%
5853   \protect\NoCaseChange
5854   {%
5855     \glsifattribute{#1}{headuc}{true}%
5856     {%
5857       \GLSxtrlong[noindex,hyper=false]{#1}[]%
5858     }%
5859     {%
5860       \glxtrlong[noindex,hyper=false]{#1}[]%
5861     }%
5862   }%
5863 }
```

`\glxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
5864 \newrobustcmd*{\glxtrtitlelong}[1]{%
5865   \glxtrlong[noindex,hyper=false]{#1}[]%
5866 }
```

`\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.

```
5867 \newcommand*{\glxtrheadlongpl}[1]{%
5868   \protect\NoCaseChange
5869   {%
5870     \glsifattribute{#1}{headuc}{true}%
5871     {%
5872       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
5873     }%
5874     {%
5875       \glxtrlongpl[noindex,hyper=false]{#1}[]%
5876     }%
5877   }%
5878 }
```

`\glxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```
5879 \newrobustcmd*{\glxtrtitlelongpl}[1]{%
5880   \glxtrlongpl[noindex,hyper=false]{#1}[]%
5881 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

5882 \newcommand*{\Glsxtrheadlong}[1]{%
5883   \protect\NoCaseChange
5884   {%
5885     \glsifattribute{#1}{headuc}{true}%
5886     {%
5887       \GLSxtrlong[noindex,hyper=false]{#1}[]%
5888     }%
5889     {%
5890       \GLSxtrlong[noindex,hyper=false]{#1}[]%
5891     }%
5892   }%
5893 }

```

Glsxtrtitlelong Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5894 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
5895   \GLSxtrlong[noindex,hyper=false]{#1}[]%
5896 }

```

lgsxtrheadlongpl Command used to display plural long form in the page header with the first letter converted to upper case.

```

5897 \newcommand*{\Glsxtrheadlongpl}[1]{%
5898   \protect\NoCaseChange
5899   {%
5900     \glsifattribute{#1}{headuc}{true}%
5901     {%
5902       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
5903     }%
5904     {%
5905       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
5906     }%
5907   }%
5908 }

```

gsxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5909 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
5910   \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
5911 }

```

\glsxtrheadfull Command used to display full form in the page header.

```

5912 \newcommand*{\glsxtrheadfull}[1]{%
5913   \protect\NoCaseChange
5914   {%
5915     \glsifattribute{#1}{headuc}{true}%
5916     {%
5917       \GLSxtrfull[noindex,hyper=false]{#1}[]%
5918     }%

```

```

5919   {%
5920     \glsxtrfull[noindex,hyper=false]{#1}[]%
5921   }%
5922 }%
5923 }

```

`\glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents.

```

5924 \newrobustcmd*{\glsxtrtitlefull}[1]{%
5925   \glsxtrfull[noindex,hyper=false]{#1}[]%
5926 }

```

`\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.

```

5927 \newcommand*{\glsxtrheadfullpl}[1]{%
5928   \protect\NoCaseChange
5929   {%
5930     \glsifattribute{#1}{headuc}{true}%
5931     {%
5932       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
5933     }%
5934     {%
5935       \glsxtrfullpl[noindex,hyper=false]{#1}[]%
5936     }%
5937   }%
5938 }

```

`\sxttrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents.

```

5939 \newrobustcmd*{\sxttrtitlefullpl}[1]{%
5940   \glsxtrfullpl[noindex,hyper=false]{#1}[]%
5941 }

```

`\Glsxtrheadfull` Command used to display full form in the page header with the first letter converted to upper case.

```

5942 \newcommand*{\Glsxtrheadfull}[1]{%
5943   \protect\NoCaseChange
5944   {%
5945     \glsifattribute{#1}{headuc}{true}%
5946     {%
5947       \GLSxtrfull[noindex,hyper=false]{#1}[]%
5948     }%
5949     {%
5950       \Glsxtrfull[noindex,hyper=false]{#1}[]%
5951     }%
5952   }%
5953 }

```

`\Glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5954 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
5955   \Glsxtrfull[noindex,hyper=false]{#1}[]%
5956 }

```

`\Glsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```

5957 \newcommand*{\Glsxtrheadfullpl}[1]{%
5958   \protect\NoCaseChange
5959   {%
5960     \glsifattribute{#1}{headuc}{true}%
5961     {%
5962       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
5963     }%
5964     {%
5965       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
5966     }%
5967   }%
5968 }

```

`\Glsxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5969 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
5970   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
5971 }

```

`\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.

```

5972 \ifdef\texorpdfstring
5973 {
5974   \newcommand*{\glsfmtshort}[1]{%
5975     \texorpdfstring
5976       {\glsxtrtitleshort{#1}}%
5977       {\glsentryshort{#1}}%
5978   }
5979 }
5980 {
5981   \newcommand*{\glsfmtshort}[1]{%
5982     \glsxtrtitleshort{#1}}
5983 }

```

Similarly for the plural version.

`\glsfmtshortpl`

```

5984 \ifdef\texorpdfstring
5985 {
5986   \newcommand*{\glsfmtshortpl}[1]{%
5987     \texorpdfstring
5988       {\glsxtrtitleshortpl{#1}}%
5989       {\glsentryshortpl{#1}}%

```



```

5990 }
5991 }
5992 {
5993   \newcommand*{\glsfmtshortpl}[1]{%
5994     \glsxtrtitleshortpl{#1}}
5995 }

```

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).

```

5996 \ifdef\texorpdfstring
5997 {
5998   \newcommand*{\Glsfmtshort}[1]{%
5999     \texorpdfstring
6000       {\Glsxtrtitleshort{#1}}%
6001       {\glsentryshort{#1}}%
6002   }
6003 }
6004 {
6005   \newcommand*{\Glsfmtshort}[1]{%
6006     \Glsxtrtitleshort{#1}}
6007 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

6008 \ifdef\texorpdfstring
6009 {
6010   \newcommand*{\Glsfmtshortpl}[1]{%
6011     \texorpdfstring
6012       {\Glsxtrtitleshortpl{#1}}%
6013       {\glsentryshortpl{#1}}%
6014   }
6015 }
6016 {
6017   \newcommand*{\Glsfmtshortpl}[1]{%
6018     \Glsxtrtitleshortpl{#1}}
6019 }

```

`\glsfmttext` As above but for the text value.

```

6020 \ifdef\texorpdfstring
6021 {
6022   \newcommand*{\glsfmttext}[1]{%
6023     \texorpdfstring
6024       {\glsxtrtitletext{#1}}%
6025       {\glsentrytext{#1}}%
6026   }
6027 }
6028 {
6029   \newcommand*{\glsfmttext}[1]{%

```

```

6030 \glstrtitletext{#1}}
6031 }

```

`\Glsfmttext` First letter converted to upper case.

```

6032 \ifdef\texorpdfstring
6033 {
6034 \newcommand*\Glsfmttext[1]{%
6035 \texorpdfstring
6036 {\Glsxtrtitletext{#1}}%
6037 {\glstrytext{#1}}%
6038 }
6039 }
6040 {
6041 \newcommand*\Glsfmttext[1]{%
6042 \Glsxtrtitletext{#1}}
6043 }

```

`\glsfmtplural` As above but for the plural value.

```

6044 \ifdef\texorpdfstring
6045 {
6046 \newcommand*\glsfmtplural[1]{%
6047 \texorpdfstring
6048 {\glstrtitleplural{#1}}%
6049 {\glstryplural{#1}}%
6050 }
6051 }
6052 {
6053 \newcommand*\glsfmtplural[1]{%
6054 \glstrtitleplural{#1}}
6055 }

```

`\Glsfmtplural` First letter converted to upper case.

```

6056 \ifdef\texorpdfstring
6057 {
6058 \newcommand*\Glsfmtplural[1]{%
6059 \texorpdfstring
6060 {\Glsxtrtitleplural{#1}}%
6061 {\glstryplural{#1}}%
6062 }
6063 }
6064 {
6065 \newcommand*\Glsfmtplural[1]{%
6066 \Glsxtrtitleplural{#1}}
6067 }

```

`\glsfmtfirst` As above but for the first value.

```

6068 \ifdef\texorpdfstring
6069 {
6070 \newcommand*\glsfmtfirst[1]{%

```

```

6071 \texorpdfstring
6072 {\glsxtrtitlefirst{#1}}}%
6073 {\glsentryfirst{#1}}}%
6074 }
6075 }
6076 {
6077 \newcommand*{\Glsfmtfirst}[1]{%
6078 \glsxtrtitlefirst{#1}}
6079 }

```

\Glsfmtfirst First letter converted to upper case.

```

6080 \ifdef\texorpdfstring
6081 {
6082 \newcommand*{\Glsfmtfirst}[1]{%
6083 \texorpdfstring
6084 {\Glsxtrtitlefirst{#1}}}%
6085 {\glsentryfirst{#1}}}%
6086 }
6087 }
6088 {
6089 \newcommand*{\Glsfmtfirst}[1]{%
6090 \Glsxtrtitlefirst{#1}}
6091 }

```

\glsfmtfirstpl As above but for the firstplural value.

```

6092 \ifdef\texorpdfstring
6093 {
6094 \newcommand*{\glsfmtfirstpl}[1]{%
6095 \texorpdfstring
6096 {\glsxtrtitlefirstplural{#1}}}%
6097 {\glsentryfirstplural{#1}}}%
6098 }
6099 }
6100 {
6101 \newcommand*{\glsfmtfirstpl}[1]{%
6102 \glsxtrtitlefirstplural{#1}}
6103 }

```

\Glsfmtfirstpl First letter converted to upper case.

```

6104 \ifdef\texorpdfstring
6105 {
6106 \newcommand*{\Glsfmtfirstpl}[1]{%
6107 \texorpdfstring
6108 {\Glsxtrtitlefirstplural{#1}}}%
6109 {\glsentryfirstplural{#1}}}%
6110 }
6111 }
6112 {
6113 \newcommand*{\Glsfmtfirstpl}[1]{%

```

```

6114 \Glsxtrtitlefirstplural{#1}}
6115 }

```

`\glsfmtlong` As above but for the long value.

```

6116 \ifdef\txorpdfstring
6117 {
6118   \newcommand*\glsfmtlong[1]{%
6119     \txorpdfstring
6120     {\glsxtrtitlelong{#1}}%
6121     {\glsentrylong{#1}}%
6122   }
6123 }
6124 {
6125   \newcommand*\glsfmtlong[1]{%
6126     \glsxtrtitlelong{#1}}
6127 }

```

`\Glsfmtlong` First letter converted to upper case.

```

6128 \ifdef\txorpdfstring
6129 {
6130   \newcommand*\Glsfmtlong[1]{%
6131     \txorpdfstring
6132     {\Glsxtrtitlelong{#1}}%
6133     {\glsentrylong{#1}}%
6134   }
6135 }
6136 {
6137   \newcommand*\Glsfmtlong[1]{%
6138     \Glsxtrtitlelong{#1}}
6139 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

6140 \ifdef\txorpdfstring
6141 {
6142   \newcommand*\glsfmtlongpl[1]{%
6143     \txorpdfstring
6144     {\glsxtrtitlelongpl{#1}}%
6145     {\glsentrylongpl{#1}}%
6146   }
6147 }
6148 {
6149   \newcommand*\glsfmtlongpl[1]{%
6150     \glsxtrtitlelongpl{#1}}
6151 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

6152 \ifdef\txorpdfstring
6153 {
6154   \newcommand*\Glsfmtlongpl[1]{%

```

```

6155     \texorpdfstring
6156     {\Glsxtrtitlelongpl{#1}}}%
6157     {\glxentrylongpl{#1}}}%
6158   }
6159 }
6160 {
6161   \newcommand*{\Glsfmtlongpl}[1]{%
6162     \Glsxtrtitlelongpl{#1}}
6163 }

```

`\glsfmtfull` In-line full format.

```

6164 \ifdef\texorpdfstring
6165 {
6166   \newcommand*{\glsfmtfull}[1]{%
6167     \texorpdfstring
6168     {\glxtrtitlefull{#1}}}%
6169     {\glxtrinlinefullformat{#1}{}}}%
6170   }
6171 }
6172 {
6173   \newcommand*{\glsfmtfull}[1]{%
6174     \glxtrtitlefull{#1}}
6175 }

```

`\Glsfmtfull` First letter converted to upper case.

```

6176 \ifdef\texorpdfstring
6177 {
6178   \newcommand*{\Glsfmtfull}[1]{%
6179     \texorpdfstring
6180     {\Glsxtrtitlefull{#1}}}%
6181     {\Glsxtrinlinefullformat{#1}{}}}%
6182   }
6183 }
6184 {
6185   \newcommand*{\Glsfmtfull}[1]{%
6186     \Glsxtrtitlefull{#1}}
6187 }

```

`\glsfmtfullpl` In-line full plural format.

```

6188 \ifdef\texorpdfstring
6189 {
6190   \newcommand*{\glsfmtfullpl}[1]{%
6191     \texorpdfstring
6192     {\glxtrtitlefullpl{#1}}}%
6193     {\glxtrinlinefullplformat{#1}{}}}%
6194   }
6195 }
6196 {
6197   \newcommand*{\glsfmtfullpl}[1]{%

```

```

6198 \glsxtrtitlefullpl{#1}}
6199 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

6200 \ifdef\teorpdfstring
6201 {
6202   \newcommand*{\Glsfmtfullpl}[1]{%
6203     \teorpdfstring
6204     {\Glsxtrtitlefullpl{#1}}%
6205     {\Glsxtrinlinefullplformat{#1}{}}%
6206   }
6207 }
6208 {
6209   \newcommand*{\Glsfmtfullpl}[1]{%
6210     \Glsxtrtitlefullpl{#1}}
6211 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

6212 \newcommand*{\RequireGlossariesExtraLang}[1]{%
6213   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
6214 }

```

`sariesExtraLang`

```

6215 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
6216   \ProvidesFile{glossariesxtr-#1.ldf}%
6217 }

```

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.)

```

6218 \@ifpackageloaded{tracklang}
6219 {%
6220   \AnyTrackedLanguages
6221   {%
6222     \ForEachTrackedDialect{\this@dialect}{%
6223       \IfTrackedLanguageFileExists{\this@dialect}%
6224       {glossariesxtr-}% prefix
6225       {.ldf}%
6226       {%
6227         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
6228       }%
6229     }%

```

```

6230      }%
6231      }%
6232  }%
6233  {}%
6234 }
6235 {}

    Load glossaries-extra-stylemods if required.
6236 \@glsxtr@redefstyles
    and set the style:
6237 \@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:

```
6238 \NeedsTeXFormat{LaTeX2e}
6239 \ProvidesPackage{glossaries-extra-stylemods}[2016/12/13 v1.08 (NLCT)]
```

Provide package options to automatically load required predefined styles. The simplest method is to just test for the existence of the file `glossary-⟨option⟩.sty`. Packages can't be loaded whilst the options are being processed, so save the list in `\@glsxtr@loadstyles`.

`sxtr@loadstyles`

```
6240 \newcommand*{\@glsxtr@loadstyles}{%
6241 \DeclareOption*{%
6242   \IfFileExists{glossary-\CurrentOption.sty}%
6243   {\eappto\@glsxtr@loadstyles{%
6244     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
6245   {\PackageError{glossaries-extra-styles}%
6246     {Unknown option '\CurrentOption'}{}}
6247 }
```

Process the package options:

```
6248 \ProcessOptions
```

Load the required packages:

```
6249 \@glsxtr@loadstyles
```

Adjust the styles that the post description hook added, but only for styles that have already been defined. All the tree styles in `glossary-tree` include the post description hook, so they don't require adjustment. Similarly for `glossary-mcols` which builds on the tree styles.

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

`ewglossarystyle`

```
6250 \providecommand{\renewglossarystyle}[2]{%
6251   \ifcsundef{@glsstyle@#1}%
6252   {%
6253     \PackageError{glossaries}{Glossary style '#1' isn't already defined}{}%

```



```

6254 }%
6255 {%
6256   \csdef{@glsstyle@#1}{#2}%
6257 }%
6258 }

```

2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying.

```

6259 \ifcsdef{@glsstyle@listdotted}
6260 {%
6261   \renewglossarystyle{listdotted}{%
6262     \setglossarystyle{list}%
6263     \renewcommand*{\glossentry}[2]{%
6264       \item[]\makebox[\glslistdottedwidth][l]{%
6265         \glsentryitem{##1}%
6266         \glstarget{##1}{\glossentryname{##1}}%
6267         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
6268         \glossentrydesc{##1}\glspostdescription}%
6269     \renewcommand*{\subglossentry}[3]{%
6270       \item[]\makebox[\glslistdottedwidth][l]{%
6271         \glssubentryitem{##2}%
6272         \glstarget{##2}{\glossentryname{##2}}%
6273         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
6274         \glossentrydesc{##2}\glspostdescription}%
6275   }
6276 }
6277 {}

```

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.

```

6278 \ifcsdef{@glsstyle@long3col}
6279 {%
6280   \renewglossarystyle{long3col}{%
6281     \renewenvironment{theglossary}%
6282       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
6283       {\end{longtable}}%
6284     \renewcommand*{\glossaryheader}{}%
6285     \renewcommand*{\glsgroupheading}[1]{}%
6286     \renewcommand{\glossentry}[2]{%
6287       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6288       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline

```

```

6289 }%
6290 \renewcommand{\subglossentry}[3]{%
6291     &
6292     \glssubentryitem{##2}%
6293     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6294     ##3\tabularnewline
6295 }%
6296 \renewcommand*{\glsgroupskip}{%
6297     \ifglsgnোগroupskip\else & &\tabularnewline\fi}%
6298 }
6299 }
6300 {}

```

Four column style:

```

6301 \ifcsdef{@glsstyle@long4col}
6302 {%
6303     \renewglossarystyle{long4col}{%
6304         \renewenvironment{theglossary}%
6305             {\begin{longtable}{llll}}%
6306             {\end{longtable}}%
6307         \renewcommand*{\glossaryheader}{}%
6308         \renewcommand*{\glsgroupheading}[1]{}%
6309         \renewcommand{\glossentry}[2]{%
6310             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6311             \glossentrydesc{##1}\glspostdescription &
6312             \glossentrysymbol{##1} &
6313             ##2\tabularnewline
6314         }%
6315         \renewcommand{\subglossentry}[3]{%
6316             &
6317             \glssubentryitem{##2}%
6318             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6319             \glossentrysymbol{##2} & ##3\tabularnewline
6320         }%
6321         \renewcommand*{\glsgroupskip}{%
6322             \ifglsgnোগroupskip\else & &\tabularnewline\fi}%
6323     }
6324 }
6325 {}

```

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.

```

6326 \ifcsdef{@glsstyle@longragged3col}
6327 {%
6328     \renewglossarystyle{longragged3col}{%

```

```

6329 \renewenvironment{theglossary}%
6330   {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}%
6331     >{\raggedright}p{\glspagelistwidth}}}%
6332   {\end{longtable}}}%
6333 \renewcommand*{\glossaryheader}{}%
6334 \renewcommand*{\glsgroupheading}[1]{}%
6335 \renewcommand{\glossentry}[2]{%
6336   \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6337   \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
6338 }%
6339 \renewcommand{\subglossentry}[3]{%
6340   &
6341   \glssubentryitem{##2}%
6342   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6343   ##3\tabularnewline
6344 }%
6345 \renewcommand*{\glsgroupskip}{%
6346   \ifglsgnogroupskip\else & \tabularnewline\fi}%
6347 }
6348 }
6349 {}

```

Four column style:

```

6350 \ifcsdef{@glstyle@altlongragged4col}
6351 {}%
6352 \renewglossarystyle{altlongragged4col}{%
6353   \renewenvironment{theglossary}%
6354     {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}l%
6355       >{\raggedright}p{\glspagelistwidth}}}%
6356     {\end{longtable}}}%
6357   \renewcommand*{\glossaryheader}{}%
6358   \renewcommand*{\glsgroupheading}[1]{}%
6359   \renewcommand{\glossentry}[2]{%
6360     \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6361     \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
6362     ##2\tabularnewline
6363   }%
6364   \renewcommand{\subglossentry}[3]{%
6365     &
6366     \glssubentryitem{##2}%
6367     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6368     \glossentrysymbol{##2} & ##3\tabularnewline
6369   }%
6370   \renewcommand*{\glsgroupskip}{%
6371     \ifglsgnogroupskip\else & & \tabularnewline\fi}%
6372 }
6373 }
6374 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment, but not the two column styles.

```

6375 \ifcsdef{@glsstyle@super3col}
6376 {%
6377   \renewglossarystyle{super3col}{%
6378     \renewenvironment{theglossary}%
6379       {\tablehead{}\tabletail{}}%
6380       \begin{supertabular}{\lp{\glsdescwidth}p{\glspagelistwidth}}}%
6381       {\end{supertabular}}}%
6382   \renewcommand*{\glossaryheader}{}%
6383   \renewcommand*{\glsgroupheading}[1]{}%
6384   \renewcommand{\glossentry}[2]{%
6385     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6386     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
6387   }%
6388   \renewcommand{\subglossentry}[3]{%
6389     &
6390     \glssubentryitem{##2}%
6391     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6392     ##3\tabularnewline
6393   }%
6394   \renewcommand*{\glsgroupskip}{%
6395     \ifglsgnpgroupskip\else & &\tabularnewline\fi}%
6396 }
6397 }
6398 {}

```

Four column styles:

```

6399 \ifcsdef{@glsstyle@super4col}
6400 {%
6401   \renewglossarystyle{super4col}{%
6402     \renewenvironment{theglossary}%
6403       {\tablehead{}\tabletail{}}%
6404       \begin{supertabular}{\llll}}}%
6405       \end{supertabular}}}%
6406   \renewcommand*{\glossaryheader}{}%
6407   \renewcommand*{\glsgroupheading}[1]{}%
6408   \renewcommand{\glossentry}[2]{%
6409     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6410     \glossentrydesc{##1}\glspostdescription &
6411     \glossentrysymbol{##1} & ##2\tabularnewline
6412   }%
6413   \renewcommand{\subglossentry}[3]{%
6414     &
6415     \glssubentryitem{##2}%
6416     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6417     \glossentrysymbol{##2} & ##3\tabularnewline
6418   }%
6419   \renewcommand*{\glsgroupskip}{%

```

```

6420 \ifglsnogroupskip\else & &\tabularnewline\fi}%
6421 }
6422 }
6423 {}

```

2.6 Super Ragged Styles

The three and four column styles require adjustment, but not the two column styles.

```

6424 \ifcsdef{@glsstyle@superragged3col}
6425 {%
6426 \renewglossarystyle{superragged3col}{%
6427 \renewenvironment{theglossary}%
6428 {\tablehead{}\tabletail}%
6429 \begin{supertabular}{1>\raggedright}p{\glsdescwidth}%
6430 >\raggedright}p{\glspagelistwidth}}}%
6431 {\end{supertabular}}}%
6432 \renewcommand*\glossaryheader{}%
6433 \renewcommand*\glsgroupheading[1]{}%
6434 \renewcommand{\glossentry}[2]{%
6435 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6436 \glossentrydesc{##1}\glspostdescription &
6437 ##2\tabularnewline
6438 }%
6439 \renewcommand{\subglossentry}[3]{%
6440 &
6441 \glssubentryitem{##2}%
6442 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6443 ##3\tabularnewline
6444 }%
6445 \renewcommand*\glsgroupskip{\ifglsnogroupskip\else &
6446 &\tabularnewline\fi}%
6447 }
6448 }
6449 {}

```

Four columns:

```

6450 \ifcsdef{@glsstyle@altsuperragged4col}
6451 {%
6452 \renewglossarystyle{altsuperragged4col}{%
6453 \renewenvironment{theglossary}%
6454 {\tablehead{}\tabletail}%
6455 \begin{supertabular}{1>\raggedright}p{\glsdescwidth}l%
6456 >\raggedright}p{\glspagelistwidth}}}%
6457 {\end{supertabular}}}%
6458 \renewcommand*\glossaryheader{}%
6459 \renewcommand{\glossentry}[2]{%
6460 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6461 \glossentrydesc{##1}\glspostdescription &
6462 \glossentrysymbol{##1} & ##2\tabularnewline

```

```

6463 }%
6464 \renewcommand{\subglossentry}[3]{%
6465     &
6466     \glssubentryitem{##2}%
6467     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6468     \glossentrysymbol{##2} & ##3\tabularnewline
6469 }%
6470 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & &
6471     &\tabularnewline\fi}%
6472 }
6473 }
6474 {}

```

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`.

```

6475 \ifdef{\@glsstyle@inline}
6476 {%
6477     \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
        Just use \glxtrpostdescription instead of \glspostdescription.
6478     \renewcommand*{\glsinlinedescformat}[3]{%
6479         \space#1\glxtrpostdescription}
6480     \renewcommand*{\glsinlinesubdescformat}[3]{%
6481         #1\glxtrpostdescription}
6482 }
6483 {}

```

2.8 Tree Styles

The `alttree` style is redefined to make it easier to made minor adjustments.

```

6484 \ifdef{\@glsstyle@alttree}
6485 {%

```

Only redefine this style if it's already been defined.

mbolDescLocation

```
\glxtraltrtreeSymbolDescLocation{\label}{\location list}
```

Layout the symbol, description and location for top-level entries.

```

6486 \newcommand{\glxtraltrtreeSymbolDescLocation}[2]{%
6487     {%
6488         \let\par\glxtrAltTreePar

```

```

6489      \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
6490      \glossentrydesc{#1}\glspostdescription \space #2\par
6491  }%
6492  }

```

`\trAltTreeIndent` Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

6493  \newlength\glxstrAltTreeIndent

```

`\lsxtrAltTreePar` Multi-paragraph descriptions need to keep the hanging indent.

```

6494  \newcommand{\glxstrAltTreePar}{%
6495    @@par
6496    \glxstrAltTreeSetHangIndent
6497    \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
6498  }

```

`\symbolDescLocation` `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

6499  \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
6500    \glxstralttreeSymbolDescLocation{#2}{#3}%
6501  }

```

`\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.

```

6502  \newlength\glxstrtreeindent

```

`\glxstralttreeInit` User-level initialisation for the alttree style.

```

6503  \newcommand*{\glxstralttreeInit}{%
6504    \settowidth{\glxstrtreeindent}{\glstreenamfmt{\glsgetwidestname\space}}%
6505    \glxstrAltTreeIndent=\parindent
6506  }

```

`\eglissetwidest` The original `\glissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

6507  \newcommand*{\eglissetwidest}[2][0]{%
6508    \protected@csedef{@glswidestname\romannumeral#1}{#2}%
6509  }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```

6510  \newcommand*{\xglissetwidest}[2][0]{%
6511    \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
6512  }

```

`\lsgetwidestname` Provide a user-level macro to obtain the widest top-level name.

```

6513  \newcommand*{\lsgetwidestname}{\@glswidestname}

```

`etwidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```
6514 \newcommand*\glsgetwidestsubname}[1]{%
6515   \ifcsundef{@glswidestname\romannumeral#1}%
6516     {\@glswidestname}%
6517     {\csuse{@glswidestname\romannumeral#1}}%
6518 }
```

`estTopLevelName` `CamelCase` is easier for long command names. Provide a `CamelCase` synonym of `\glsfindwidesttoplevelname`

```
6519 \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.

```
6520 \newrobustcmd*\glsFindWidestUsedTopLevelName}[1][\@glo@types]{%
6521   \dimen@=0pt\relax
6522   \gls@tmplen=0pt\relax
6523   \foralllglossaries[#1]{\@gls@type}%
6524   {%
6525     \forglsentries[\@gls@type]{\@glo@label}%
6526     {%
6527       \ifglsused{\@glo@label}%
6528       {%
6529         \ifglsashasparent{\@glo@label}%
6530         {}%
6531         {%
6532           \settowidth{\dimen@}%
6533             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
6534           \ifdim\dimen@>\gls@tmplen
6535             \gls@tmplen=\dimen@
6536             \eglssetwidest{\glsentryname{\@glo@label}}%
6537           \fi
6538         }%
6539       }%
6540     }%
6541   }%
6542 }%
6543 }
```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```
6544 \newrobustcmd*\glsFindWidestUsedAnyName}[1][\@glo@types]{%
6545   \dimen@=0pt\relax
6546   \gls@tmplen=0pt\relax
6547   \foralllglossaries[#1]{\@gls@type}%
6548   {%
6549     \forglsentries[\@gls@type]{\@glo@label}%
6550     {%
```



```

6551     \ifglused{\@glo@label}%
6552     {%
6553         \settowidth{\dimen@}%
6554         {\glstreenamefmt{\glentryname{\@glo@label}}}%
6555         \ifdim\dimen@>\gls@tmplen
6556             \gls@tmplen=\dimen@
6557             \eglssetwidest{\glentryname{\@glo@label}}%
6558         \fi
6559     }%
6560     {%
6561     }%
6562 }%
6563 }

```

ndWidestAnyName Like the above but doesn't check if the entry has been used.

```

6564 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
6565     \dimen@=0pt\relax
6566     \gls@tmplen=0pt\relax
6567     \forallglossaries[#1]{\@gls@type}%
6568     {%
6569         \forglentries[\@gls@type]{\@glo@label}%
6570         {%
6571             \settowidth{\dimen@}%
6572             {\glstreenamefmt{\glentryname{\@glo@label}}}%
6573             \ifdim\dimen@>\gls@tmplen
6574                 \gls@tmplen=\dimen@
6575                 \eglssetwidest{\glentryname{\@glo@label}}%
6576             \fi
6577         }%
6578     }%
6579 }

```

estUsedLevelTwo This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

6580 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
6581     \dimen@=0pt\relax
6582     \dimen@i=0pt\relax
6583     \dimen@ii=0pt\relax
6584     \forallglossaries[#1]{\@gls@type}%
6585     {%
6586         \forglentries[\@gls@type]{\@glo@label}%
6587         {%
6588             \ifglused{\@glo@label}%
6589             {%
6590                 \ifglshasparent{\@glo@label}%
6591                 {%
6592                     \edef\@glo@parent{\csuse{glo@glsetoklabel}{\@glo@label}@parent}}%
6593                     \ifglshasparent{\@glo@parent}%
6594                 }%

```

```

6595         \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
6596         \ifglshasparent{\@glo@parent}%
6597         {%
6598         {%
6599             \settowidth{\gls@tmplen}%
6600             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6601             \ifdim\gls@tmplen>\dimen@ii
6602             \dimen@ii=\gls@tmplen
6603             \eglssetwidest[2]{\glsentryname{\@glo@label}}%
6604             \fi
6605         }%
6606     }%
6607     {%
6608         \settowidth{\gls@tmplen}%
6609         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6610         \ifdim\gls@tmplen>\dimen@i
6611         \dimen@i=\gls@tmplen
6612         \eglssetwidest[1]{\glsentryname{\@glo@label}}%
6613         \fi
6614     }%
6615 }%
6616 {%
6617     \settowidth{\gls@tmplen}%
6618     {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6619     \ifdim\gls@tmplen>\dimen@
6620     \dimen@=\gls@tmplen
6621     \eglssetwidest{\glsentryname{\@glo@label}}%
6622     \fi
6623 }%
6624 }%
6625 {}%
6626 }%
6627 }%
6628 }

```

dWidestLevelTwo This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

6629 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
6630     \dimen@=0pt\relax
6631     \dimen@i=0pt\relax
6632     \dimen@ii=0pt\relax
6633     \foralllglossaries[#1]{\@gls@type}%
6634     {%
6635         \forglsentries[\@gls@type]{\@glo@label}%
6636         {%
6637             \ifglshasparent{\@glo@label}%
6638             {%
6639                 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
6640                 \ifglshasparent{\@glo@parent}%
6641                 {%

```

```

6642      \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
6643      \ifglshasparent{\@glo@parent}%
6644      {}%
6645      {%
6646        \settowidth{\gls@tmplen}%
6647          {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6648        \ifdim\gls@tmplen>\dimen@ii
6649          \dimen@ii=\gls@tmplen
6650          \eglssetwidest[2]{\glsentryname{\@glo@label}}%
6651        \fi
6652      }%
6653    }%
6654    {%
6655      \settowidth{\gls@tmplen}%
6656        {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6657      \ifdim\gls@tmplen>\dimen@i
6658        \dimen@i=\gls@tmplen
6659        \eglssetwidest[1]{\glsentryname{\@glo@label}}%
6660      \fi
6661    }%
6662  }%
6663  {%
6664    \settowidth{\gls@tmplen}%
6665      {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6666    \ifdim\gls@tmplen>\dimen@
6667      \dimen@=\gls@tmplen
6668      \eglssetwidest{\glsentryname{\@glo@label}}%
6669    \fi
6670  }%
6671 }%
6672 }%
6673 }

```

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.

```

6674 \newrobustcmd*\glsFindWidestUsedAnyNameSymbol[2][\@glo@types]{%
6675   \dimen@=0pt\relax
6676   \gls@tmplen=0pt\relax
6677   #2=0pt\relax
6678   \forallglossaries[#1]{\@gls@type}%
6679   {%
6680     \forglsentries[\@gls@type]{\@glo@label}%
6681     {%
6682       \ifglused{\@glo@label}%
6683       {%
6684         \settowidth{\dimen@}%
6685           {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6686         \ifdim\dimen@>\gls@tmplen
6687           \gls@tmplen=\dimen@

```

```

6688         \eglssetwidest{\glsentryname{\@glo@label}}}%
6689         \fi
6690         \settowidth{\dimen@}%
6691         {\glsentrysymbol{\@glo@label}}}%
6692         \ifdim\dimen@>#2\relax
6693             #2=\dimen@
6694         \fi
6695     }%
6696     {}%
6697 }%
6698 }%
6699 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

6700 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
6701     \dimen@=0pt\relax
6702     \gls@tmplen=0pt\relax
6703     #2=0pt\relax
6704     \forallglossaries[#1]{\@gls@type}%
6705     {%
6706         \forglsentries[\@gls@type]{\@glo@label}%
6707         {%
6708             \settowidth{\dimen@}%
6709             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
6710             \ifdim\dimen@>\gls@tmplen
6711                 \gls@tmplen=\dimen@
6712                 \eglssetwidest{\glsentryname{\@glo@label}}}%
6713             \fi
6714             \settowidth{\dimen@}%
6715             {\glsentrysymbol{\@glo@label}}}%
6716             \ifdim\dimen@>#2\relax
6717                 #2=\dimen@
6718             \fi
6719         }%
6720     }%
6721 }

```

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.

```

6722 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
6723     \dimen@=0pt\relax
6724     \gls@tmplen=0pt\relax
6725     #2=0pt\relax
6726     #3=0pt\relax
6727     \forallglossaries[#1]{\@gls@type}%
6728     {%
6729         \forglsentries[\@gls@type]{\@glo@label}%

```

```

6730     {%
6731         \ifglsused{\@glo@label}%
6732     {%
6733         \settowidth{\dimen@}%
6734             {\glstreenamefmt{\glentryname{\@glo@label}}}%
6735         \ifdim\dimen@>\gls@tmplen
6736             \gls@tmplen=\dimen@
6737         \eglssetwidest{\glentryname{\@glo@label}}%
6738         \fi
6739         \settowidth{\dimen@}%
6740             {\glentrysymbol{\@glo@label}}%
6741         \ifdim\dimen@>#2\relax
6742             #2=\dimen@
6743         \fi
6744         \settowidth{\dimen@}%
6745             {\GlsXtrFormatLocationList{\glentrynumberlist{\@glo@label}}}%
6746         \ifdim\dimen@>#3\relax
6747             #3=\dimen@
6748         \fi
6749     }%
6750 }%
6751 }%
6752 }%
6753 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

6754 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
6755     \dimen@=0pt\relax
6756     \gls@tmplen=0pt\relax
6757     #2=0pt\relax
6758     #3=0pt\relax
6759     \foralllglossaries[#1]{\@gls@type}%
6760     {%
6761         \forglsentries[\@gls@type]{\@glo@label}%
6762         {%
6763             \settowidth{\dimen@}%
6764                 {\glstreenamefmt{\glentryname{\@glo@label}}}%
6765             \ifdim\dimen@>\gls@tmplen
6766                 \gls@tmplen=\dimen@
6767             \eglssetwidest{\glentryname{\@glo@label}}%
6768             \fi
6769             \settowidth{\dimen@}%
6770                 {\glentrysymbol{\@glo@label}}%
6771             \ifdim\dimen@>#2\relax
6772                 #2=\dimen@
6773             \fi
6774             \settowidth{\dimen@}%
6775                 {\GlsXtrFormatLocationList{\glentrynumberlist{\@glo@label}}}%
6776             \ifdim\dimen@>#3\relax

```

```

6777         #3=\dimen@
6778         \fi
6779     }%
6780 }%
6781 }

```

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.

```

6782 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
6783     \dimen@=0pt\relax
6784     \gls@tmplen=0pt\relax
6785     #2=0pt\relax
6786     \forallglossaries[#1]{\@gls@type}%
6787     {%
6788         \forallglsentries[\@gls@type]{\@glo@label}%
6789         {%
6790             \ifglsused{\@glo@label}%
6791             {%
6792                 \settowidth{\dimen@}%
6793                 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
6794                 \ifdim\dimen@>\gls@tmplen
6795                     \gls@tmplen=\dimen@
6796                     \eglssetwidest{\glsentryname{\@glo@label}}%
6797                 \fi
6798                 \settowidth{\dimen@}%
6799                 {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
6800                 \ifdim\dimen@>#2\relax
6801                     #2=\dimen@
6802                 \fi
6803             }%
6804         }%
6805     }%
6806 }%
6807 }

```

AnyNameLocation Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

6808 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
6809     \dimen@=0pt\relax
6810     \gls@tmplen=0pt\relax
6811     #2=0pt\relax
6812     \forallglossaries[#1]{\@gls@type}%
6813     {%
6814         \forallglsentries[\@gls@type]{\@glo@label}%
6815         {%
6816             \settowidth{\dimen@}%
6817             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
6818             \ifdim\dimen@>\gls@tmplen
6819                 \gls@tmplen=\dimen@

```

```

6820         \eglssetwidest{\glsentryname{\@glo@label}}}%
6821     \fi
6822     \settowidth{\dimen@}%
6823     {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
6824     \ifdim\dimen@>\#2\relax
6825         #2=\dimen@
6826     \fi
6827 }%
6828 }%
6829 }

```

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`.

```

6830 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
6831     \glstreeindent=\glsxtrtreetopindent\relax
6832 }

```

teTreeSubIndent

```

6833 %\cs{\glsxtrComputeTreeSubIndent}\marg{level}\marg{label}\marg{register}
6834 %\end{macrocode}
6835 % Compute the indent for the sub-entries. The first argument is the
6836 % level, the second argument is the entry label and the third
6837 % argument is the length register used to store the computed indent.
6838 % \begin{macrocode}
6839 \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
6840     \ifcsundef{@glswidestname\romannumeral#1}%
6841     {%
6842         \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
6843     }%
6844     {%
6845         \settowidth{#3}{\glstreenamefmt{%
6846             \csname @glswidestname\romannumeral#1\endcsname\space}}%
6847     }%
6848 }

```

eeSetHangIndent Set `\hangindent` for top-level entries:

```

6849 \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set `\hangindent` for sub-entries:

```

6850 \newcommand*{\glsxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine `alttree`:

```

6851 \renewglossarystyle{alttree}{%
6852     \renewenvironment{theglossary}%
6853     {%
6854         \glsxtralttreeInit
6855         \def\@gls@prevlevel{-1}%

```

```

6856     \mbox{}\par}%
6857     {\par}%
6858 \renewcommand*{\glossaryheader}{}%
6859 \renewcommand*{\glsgroupheading}[1]{}%
6860 \renewcommand{\glossentry}[2]{%
6861     \ifnum\@gls@prevlevel=0\relax
6862     \else
6863         \glstrComputeTreeIndent{##1}%
6864     \fi
6865     \parindent\glstreeindent
6866     \glstrAltTreeSetHangIndent
6867     \makebox[0pt][r]%
6868     {%
6869         \glstreenamebox{\glstreeindent}%
6870         {%
6871             \glstryitem{##1}%
6872             \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
6873         }%
6874     }%
6875     \glstralttreeSymbolDescLocation{##1}{##2}%
6876     \def\@gls@prevlevel{0}%
6877 }
6878 \renewcommand{\subglossentry}[3]{%
6879     \ifnum##1=1\relax
6880         \glssubentryitem{##2}%
6881     \fi
6882     \ifnum\@gls@prevlevel=##1\relax
6883     \else
6884         \glstrComputeTreeSubIndent{##1}{##2}{\@gls@tmplen}%
6885         \ifnum\@gls@prevlevel<##1\relax
6886             \setlength\glstreeindent\@gls@tmplen
6887             \addtolength\glstreeindent\parindent
6888             \parindent\glstreeindent
6889         \else
6890             \ifnum\@gls@prevlevel=0\relax
6891                 \glstrComputeTreeIndent{##2}%
6892             \else
6893                 \glstrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
6894             \fi
6895             \addtolength\parindent{-\glstreeindent}%
6896             \setlength\glstreeindent\parindent
6897         \fi
6898     \fi
6899     \glstrAltTreeSetSubHangIndent{##1}%
6900     \makebox[0pt][r]{\glstreenamebox{\@gls@tmplen}{%
6901         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
6902     \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
6903     \def\@gls@prevlevel{##1}%
6904 }%

```



```

6905     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
6906   }
6907 }%
6908 {%

```

Assume the style isn't required if it hasn't already been defined.

```

6909 }

```

Reset the default style

```

6910 \ifx\@glossary@default@style\relax
6911 \else
6912   \setglossarystyle{\@glxtr@current@style}
6913 \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)	
General: Initial experimental release	4
0.2 (2015-11-30)	
\Glsfmtshort: new	177
\glsfmtshort: new	176
\Glsfmtshortpl: new	177
\glsfmtshortpl: new	176
short: switched inline full form to short (long)	143
0.3 (2015-12-02)	
\@ACRlong: added redefinition	40
\@ACRlongpl: added redefinition	41
\@ACRshort: added redefinition	38
\@ACRshortpl: added redefinition	39
\@Acrlong: added redefinition	40
\@Acrlongpl: added redefinition	41
\@Acrshort: added redefinition	38
\@Acrshortpl: added redefinition	39
\@GLSdesc: added redefinition	34
\@GLSdescplural@: added redefinition	34
\@GLSfirst@: added redefinition	31
\@GLSfirstplural@: added redefinition	33
\@GLSname@: added redefinition	33
\@GLSplural@: added redefinition	32
\@GLSSymbol@: added redefinition	35
\@GLSSymbolplural@: added redefinition	35
\@GLStext@: added redefinition	30
\@GLSuseri@: added redefinition	36
\@GLSuserii@: added redefinition	36
\@GLSuseriii@: added redefinition	36
\@GLSuseriv@: added redefinition	37
\@GLSuseriv@: added redefinition	37
\@GLSuseriv@: added redefinition	37
\@GLSdesc@: added redefinition	34
\@GLSdescplural@: added redefinition	34
\@GLSfirst@: added redefinition	31
\@GLSfirstplural@: added redefinition	33
\@GLSname@: added redefinition	33
\@GLSplural@: added redefinition	32
\@GLSSymbol@: added redefinition	35
\@GLSSymbolplural@: added redefinition	35
\@GLStext@: added redefinition	31
\@GLSuseri@: added redefinition	36
\@GLSuserii@: added redefinition	36
\@GLSuseriii@: added redefinition	36
\@GLSuseriv@: added redefinition	36
\@GLSuseriv@: added redefinition	37
\@GLSuseriv@: added redefinition	37
\@GLSuseriv@: added redefinition	37
\@GLSdesc@: added redefinition	34
\@GLSdescplural@: added redefinition	34
\@GLSfirst@: added redefinition	31
\@GLSfirstplural@: added redefinition	33
\@GLSname@: added redefinition	33
\@GLSplural@: added redefinition	32
\@GLSSymbol@: added redefinition	35
\@GLSSymbolplural@: added redefinition	35
\@GLStext@: added redefinition	31
\@GLSuseri@: added redefinition	36
\@GLSuserii@: added redefinition	36
\@GLSuseriii@: added redefinition	36
\@GLSuseriv@: added redefinition	36
\@GLSuseriv@: added redefinition	37
\@GLSuseriv@: added redefinition	37
\@GLSdesc@: added redefinition	34
\@GLSdescplural@: added redefinition	34
\@GLSfirst@: added redefinition	31
\@GLSfirstplural@: added redefinition	32
\@GLSplural@: added redefinition	32
\@GLSSymbol@: added redefinition	35
\@GLSxtr@defaultnoglossarywarning: new	79
\@GLSxtr@field@linkdefs: new	30
\@GLSxtr@insertdots: new	116
\@print@glossary: added redefinition	76
\glsabbrvdefaultfont: renamed from \abbrvdefaultfont	120
\glsaccessdesc: new	86
\glsaccessdescplural: new	86
\glsaccessfirst: new	83
\glsaccessfirstplural: new	84
\glsaccesslong: new	88
\glsaccesslong: new	88
\glsaccessname: new	82
\glsaccessplural: new	83
\glsaccessshort: new	87
\glsaccessshort: new	87
\glsaccessshortpl: new	88

\glsaccesssshortpl: new	88	\cGLSpl: new	62
\glsaccessssymbol: new	85	\cGLSpl@: new	62
\glsaccessssymbolplural: new	85	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	82	new	57
\glsentryfmt: added check for short ..	27	\cGLS: new	62
\glslongpltok: new	116	\cGLSformat: new	62
\glsshortpltok: new	116	\cGLSpl: new	62
\glsxtrdiscardperiod: added check		\cGLSplformat: new	62
for plural	113	\GlossariesExtraWarningNoLine:	
\GLSxtrlongpl: new	129	new	9
\Glsxtrlongpl: new	128	\glsenableentrycount: new	58
\glsxtrlongpl: new	128	\glsfirstabbrvdefaultfont: new ..	119
\glsxtrNoGlossaryWarning: new	12	\glsfirstlongdefaultfont: new ...	120
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirst: new	179
new	112	\glsfmtfirst: new	178
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtfirstpl: new	179
new	112	\glsfmtfirstpl: new	179
\glsxtrpostlinkendsentence: new ..	112	\Glsfmtplural: new	178
\GLSxtrshortpl: new	127	\glsfmtplural: new	178
\Glsxtrshortpl: new	127	\Glsfmtshort: changed to use	
\glsxtrshortpl: new	126	\Glsxtrtitleshort	177
short-long-desc: fixed name to use		renamed from \Glsentryfmtshort ..	177
\glslabeltok	138	\glsfmtshort: changed to use	
\newabbreviation: fixed family name in		\glsxtrtitleshort	176
\setkeys	116	renamed from \glsentryfmtshort ..	176
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	136	\Glsxtrtitleshortpl	177
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glsentryfmtshortpl	177
redefinition of \acronymtype	10	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	176
\Glsxtrshort	177	renamed from	
\glsfmtshort: changed to use		\glsentryfmtshortpl	176
\glsxtrshort	176	\Glsfmttext: new	178
\Glsfmtshortpl: changed to use		\glsfmttext: new	177
\glsxtrshortpl	177	\glshasattribute: new	94
\glsfmtshortpl: changed to use		\glshascategoryattribute: new	93
\glsxtrshortpl	176	\GlsXtrEnableEntryCounting: new ..	57
\glsxtrifemptyglossary: new	15	\glsxtrifcounttrigger: new	60
\glsxtrnewnumber: added extra		\glsxtrscfont: new	147
argument	97	\glsxtrscsuffix: new	148
\glsxtrnewsymbol: added extra		\glsxtrsmfont: new	151
argument	97	\glsxtrsmsuffix: new	152
\MakeAcronymsAbbreviations: set the		short-em: new	158
default type to \acronymtype	71	short-em-desc: new	159
\newterm: fixed name argument	96	short-em-footnote: new	160
0.5 (2015-12-07)		short-em-long: new	157
\cGLS: new	62	short-em-long-desc: new	157
\cGLS@: new	62	short-em-postfootnote: new	161

short-sc-footnote: new	151	\Glsxtrheadtext: now uses headuc	
short-sc-postfootnote: new	151	attribute	170
short-sm: new	153	\glxtrheadtext: now uses headuc	
short-sm-desc: new	153	attribute	169
short-sm-footnote: new	154	short-long: switch off regular attribute	
short-sm-long: new	152	if set	137
short-sm-long-desc: new	152	short-long-desc: switch off regular	
short-sm-postfootnote: new	154	attribute if set	138
long-noshort-em: new	159	long-short: switch off regular attribute	
long-noshort-em-desc: new	160	if set	135
long-noshort-sm: new	153	long-short-desc: switch off regular	
long-noshort-sm-desc: new	154	attribute if set	137
long-short-em: new	155	footnote: switch off regular attribute if	
long-short-em-desc: new	155	set	139
long-short-sm: new	152	postfootnote: switch off regular	
long-short-sm-desc: new	152	attribute if set	141
0.5.1 (2015-12-02)		0.5.2 (2015-12-08)	
\Glsaccessstext: new	83	\@GLSdesc@: added accessibility support	34
0.5.1 (2015-12-07)		\@GLSdescplural@: added accessibility	
\@glxtr@doaccsupp: new	12	support	34
General: removed \ifglxtruseuhead	168	\@GLSfirst@: added accessibility	
\Glsaccessdesc: new	86	support	31
\Glsaccessdescplural: new	87	\@GLSfirstplural@: added accessibility	
\Glsaccessfirst: new	84	support	33
\Glsaccessfirstplural: new	84	\@GLSname@: added accessibility support	33
\Glsaccessname: new	82	\@GLSplural@: added accessibility	
\Glsaccessplural: new	83	support	32
\Glsaccesssymbol: new	85	\@GLSsymbol@: added accessibility	
\Glsaccesssymbolplural: new	85	support	35
\Glsxtrheadfirst: now uses headuc		\@GLSsymbolplural@: added	
attribute	171	accessibility support	35
\glxtrheadfirst: now uses headuc		\@GLStext@: added accessibility support	30
attribute	171	\@GLSdesc@: added accessibility support	34
\Glsxtrheadfirstplural: now uses		\@GLSdescplural@: added accessibility	
headuc attribute	172	support	34
\glxtrheadfirstplural: now uses		\@GLSfirst@: added accessibility	
headuc attribute	172	support	31
\Glsxtrheadplural: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	171	support	33
\glxtrheadplural: now uses headuc		\@GLSname@: add accessibility support ..	33
attribute	170	\@GLSplural@: added accessibility	
\Glsxtrheadshort: now uses headuc		support	32
attribute	169	\@GLSsymbol@: added accessibility	
\glxtrheadshort: now uses headuc		support	35
attribute	168	\@GLSsymbolplural@: added	
\Glsxtrheadshortpl: now uses headuc		accessibility support	35
attribute	169	\@GLStext@: added accessibility support	31
\glxtrheadshortpl: now uses headuc		\@GLSdesc@: added accessibility support	34
attribute	168		

\@glsdescplural@: added accessibility support	34	\glsxtrnewabbrevpresetkeyhook: new	118
\@glsfirst@: added accessibility support	31	\glsxtrtagfont: new	110
\@glsfirstplural@: added accessibility support	32	\KV@printgloss@nonumberlist: added	27
\@glsname@: added accessibility support	33	\mfu@checkword@do: added	110
\@glsplural@: added accessibility support	32	\setabbreviationstyle: added check for post-definition style switch	132
\@glsymbol@: added accessibility support	35	0.5.3 (2015-12-09)	
\@glsymbolplural@: added accessibility support	35	\@glsxtr@autoindex@at: new	106
\@glstext@: added accessibility support	30	\@glsxtr@autoindex@encap: new ...	106
\@glsxtr@activate@initialtagging: new	110	\@glsxtr@autoindex@esc: new	107
\@glsxtr@do@titlecaps@warn: new	110	\@glsxtr@autoindex@level: new ...	107
\@glsxtr@tag: new	110	\@glsxtr@autoindex@setname: new	105
General: fixed typo in glossaries-accsupp and tidied up code to use just one		\@glsxtr@doabbreviationsdef: new ..	9
\@ifpackageloaded	82	General: removed	
removed \glsxtrabbrvfmt	130	\GlsXtrNoGlsWarningNoAutoMakeMain	78
\glossaryentrynumbers: added	24	\glsdescwidth: added	24
\Glossentrydesc: added	108	\glspagelistwidth: added	24
\Glossentryname: added	102	\glsxtrdoautoindexname: new	104
\Glossentrysymbol: added	109	\glsxtrpostnamehook: new	104
\glossentrysymbol: added	108	\if@glsxtr@format@override: new	104
\GLSaccessdesc: new	86, 91	\ProvidesGlossariesExtraLang: new	182
\GLSaccessdescplural: new	87, 91	\RequireGlossariesExtraLang: new	182
\GLSaccessfirst: new	84, 90	0.5.4 (2015-12-15)	
\GLSaccessfirstplural: new	84, 90	\@newglossaryentry@defunitcounters: new	63
\GLSaccesslong: new	88, 92	\@GLSxtr@p@acrlong@: new	50
\GLSaccesslongpl: new	89, 92	\@GLSxtr@p@acrlongpl@: new	51
\Glsaccesslongpl: new	89	\@GLSxtr@p@acrshort@: new	50
\glsaccesslongpl: new	89	\@GLSxtr@p@acrshortpl@: new	50
\GLSaccessname: new	82, 89	\@GLSxtr@p@long@: new	50
\GLSaccessplural: new	83, 90	\@GLSxtr@p@longpl@: new	50
\GLSaccessshort: new	87, 91	\@GLSxtr@p@plural@: new	48
\GLSaccessshortpl: new	88, 92	\@GLSxtr@p@short@: new	49
\GLSaccesssymbol: new	85, 90	\@GLSxtr@p@shortpl@: new	49
\GLSaccesssymbolplural: new ...	86, 91	\@GLSxtr@p@text@: new	48
\GLSaccessstext: new	83, 90	\@GlsXtrEnableOnTheFly: new	20
\glsentryfmt: moved		\@Glsxtr: new	21
\glssetabbrvfmt from		\@Glsxtr@p@acrlong@: new	50
\glsxtrabbrvfmt to here	27	\@Glsxtr@p@acrlongpl@: new	51
\GlsXtrEnableInitialTagging: new	109	\@Glsxtr@p@acrshort@: new	50
\glsxtrfieldtitlecase: new	98	\@Glsxtr@p@acrshortpl@: new	50
\GlsXtrFormatLocationList: new ...	25	\@Glsxtr@p@long@: new	50
		\@Glsxtr@p@longpl@: new	50
		\@Glsxtr@p@plural@: new	48
		\@Glsxtr@p@short@: new	49
		\@Glsxtr@p@shortpl@: new	49
		\@Glsxtr@p@text@: new	48

\@Glsxtrpl: new	22	\glxstr: new	21
\@alt@glshyp@opt: new	46	\glxstrcat: new	20
\@glscalt@hyp@opt: new	46	\glxstrdowrglossaryhook: new	46
\@glscalt@hyp@opt@char: new	46	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	46	new	68
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new	20
new	64	\Glsxtrpl: new	22
\@glsc@local@increment@currunitcount:		\glxtrpl: new	21
new	64	\glxtrpostlocalreset: new	57
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new	56
new	44	\glxtrpostreset: new	57
\@glxstr: new	21	\glxtrpostunset: new	56
\@glxstr@addunitcounter: new	63	\glxtrprotectlinks: new	47
\@glxstr@currunitcount: new	65	\GlsXtrSetAltModifier: new	46
\@glxstr@ifunitcounter: new	64	\GlsXtrSetDefaultGlsOpts: new	45
\@glxstr@p@acrlong@: new	50	\glxtrstarflywarn: new	20
\@glxstr@p@acrlongpl@: new	51	\GlsXtrWarning: new	22
\@glxstr@p@acrshort@: new	50	\MakeAcronymsAbbreviations: now	
\@glxstr@p@acrshortpl@: new	50	disables \setacronymstyle	71
\@glxstr@p@long@: new	49	1.0 (2016-01-24)	
\@glxstr@p@longpl@: new	50	\@glxstr@autoindexcrossrefs: new ..	9
\@glxstr@p@plural@: new	48	\@glxstr@idx@displaynumberlist:	
\@glxstr@p@short@: new	48	new	74
\@glxstr@p@shortpl@: new	49	\@glxstr@idx@entrynumberlist: new	76
\@glxstr@p@text@: new	48	\@glxstr@noidx@displaynumberlist:	
\@glxstr@prevunitcount: new	65	new	75
\@glxstr@setentryunitcountunsetattr:		\@glxstr@noidx@entrynumberlist:	
new	69	new	76
\@glxstr@unitcountlist: new	63	\@glxstr@noidx@numberlistloop:	
\@glxstrpl: new	21	new	75
\@newglossaryentryposthook: added		\@glxstr@reg@glosslist: new	72
empty see value if not set and added		\makeglossaries: new	72
‘see’ to field key map	17	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	20	\glxtrdiscardperiod: added check	
\cGlsformat: added	63	for first use	113
\cGlsformat: added	63	short-desc: fixed typo in	
\cGlsplformat: added	63	\glxtrinlinefullformat and	
\cGlsplformat: added	63	added missing second argument ...	144
\glsdisablehyper: added	47	1.02 (2016-04-25)	
\glsdohyperlink: added	47	\@glxstr@current@style: new	23
\glsdonohyperlink: added	47	\Glsfmtfull: new	181
\glsenableentryunitcount: new	65	\glsfmtfull: new	181
\glshasattribute: added check for		\Glsfmtfullpl: new	182
entry’s existence	94	\glsfmtfullpl: new	181
\glusifattribute: added check for		\Glsfmtlong: new	180
entry’s existence	94	\glsfmtlong: new	180
\glspostlinkhook: added existence		\Glsfmtlongpl: new	180
check	111	\glsfmtlongpl: new	180
\Glsxtr: new	21	\Glsxtrheadfull: new	175

\glxtrheadfull: new	174	\@GLSplural@: set abbreviation and regular format	32
\Glsxtrheadfullpl: new	176	\@GLSsymbol@: set regular format	35
\glxtrheadfullpl: new	175	\@GLSsymbolplural@: set regular format	35
\glxtrheadlong: new	173	\@GLStext@: set abbreviation and regular format	30
\Glsxtrheadlongpl: new	174	\@GLSuseri@: set regular format	36
\glxtrheadlongpl: new	173	\@GLSuserii@: set regular format	36
\Glsxtrtitlefull: new	175	\@GLSuseriii@: set regular format	36
\glxtrtitlefull: new	175	\@GLSuseriv@: set regular format	37
\Glsxtrtitlefullpl: new	176	\@GLSuserv@: set regular format	37
\glxtrtitlefullpl: new	175	\@GLSuservi@: set regular format	37
\Glsxtrtitlelong: new	174	\@Glsdesc@: set abbreviation and regular format	34
\glxtrtitlelong: new	173	\@Glsdescplural@: set abbreviation and regular format	34
\Glsxtrtitlelongpl: new	174	\@Glsfirst@: set abbreviation and regular format	31
\glxtrtitlelongpl: new	173	\@Glsfirstplural@: set abbreviation and regular format	33
\ifglxtrinsetinside: new	135	\@Glsname@: set abbreviation and regular format	33
postfootnote: added redef of \glxtrsetupfulldefs	141	\@Glsplural@: set abbreviation and regular format	32
stylemods: new	13	\@Glsymbol@: set regular format	35
1.03 (2016-04-27)		\@Glsymbolplural@: set regular format	35
\@GLSfirstplural@: bug fix: misspelt cs name	33	\@Glstext@: set abbreviation and regular format	31
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	32	\@GLSuseri@: set regular format	36
\@Glsfirstplural@: bug fix: misspelt cs name	33	\@GLSuserii@: set regular format	36
\@Glsplural@: fixed bug \@Glsplural@ should be redefined not \@Glsplural	32	\@GLSuseriii@: set regular format	36
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	32	\@GLSuseriv@: set regular format	36
\glxtrtitlelongpl: bug fix: changed \glxtrlong to \glxtrlongpl ..	173	\@GLSuserv@: set regular format	37
\glxtrtitleshortpl: bug fix: changed \glxtrshort to \glxtrshortpl ..	168	\@GLSuservi@: set regular format	37
1.04 (2015-04-30)		\@gls@preglossaryhook: added check for entry's existence	111
short-em-footnote: renamed from "footnote-em"	160	\@Glsdesc@: set abbreviation and regular format	34
1.04 (2016-05-02)		\@Glsdescplural@: set abbreviation and regular format	34
\@@glxtrpostloctag: new	26	\@Glsfirst@: set abbreviation and regular format	31
\@GLSdesc@: set abbreviation and regular format	34	\@Glsfirstplural@: set abbreviation and regular format	32
\@GLSdescplural@: set abbreviation and regular format	34	\@Glsname@: set abbreviation and regular format	33
\@GLSfirst@: set abbreviation format ..	31	\@Glsplural@: set abbreviation and regular format	32
\@GLSfirstplural@: set abbreviation and regular format	33	\@glsymbol@: set regular format	35
\@GLSname@: set abbreviation and regular format	33		

\@glssymbolplural@: set regular format	35	short-em-nolong-desc: new	159
\@glstext@: set abbreviation and regular format	30	short-em-postfootnote: renamed from “postfootnote-em”	161
\@glstr@deprecated@abbrstyle: new	134	short-footnote: new	141
\@glstr@do@style: new	13	short-long-user: new	163
\@glstr@doloctag: new	27	short-long-user-desc: new	164
\@glstr@idx@entrynumberlist: switched from \let to \newcommand	76	short-nolong: new	144
\@glstr@pagetag: new	26	short-nolong-desc: new	145
\@glstr@pagetag: new	26	short-postfootnote: new	142
\@glstr@preloctag: new	26	short-sc-footnote: renamed from “footnote-sc”	151
\@glstr@postloctag: new	26	short-sc-nolong: new	149
\@glstr@preloctag: new	26	short-sc-nolong-desc: new	150
\glossentrydesc: added glossdescfont attribute check	98	short-sc-postfootnote: renamed from “postfootnote-sc”	151
\Glossentryname: added glossnamefont attribute check	102	short-sm-footnote: renamed from “footnote-sm”	154
\glossentryname: added glossnamefont attribute check	100	short-sm-nolong: new	153
moved post name hook inside condition	102	short-sm-nolong-desc: new	153
\glsabbrvmfont: new	155	short-sm-postfootnote: renamed from “postfootnote-sm”	154
\glsabbrvuserfont: new	162	\letabbreviationstyle: new	134
\glsfirstabbrvmfont: new	155	\newabbreviationstyle: bug fix: corrected test for existence	133
\glsfirstabbrvuserfont: new	162	long-em-noshort-em: new	159
\glsfirstlongemfont: new	155	long-em-noshort-em-desc: new	160
\glsfirstlonguserfont: new	162	long-em-short-em: new	156
\glsifnotregularcategory: new	95	long-em-short-em-desc: new	156
\glslongdefaultfont: new	120	long-noshort: new	147
\glslongemfont: new	155	long-noshort-desc: new	147
\glslongfont: new	120	long-noshort-em: renamed from “long-em”	159
\glslonguserfont: new	162	long-noshort-em-desc: renamed from “long-desc-em”	160
\glstrassignfieldfont: new	30	long-noshort-sc: renamed from “long-sc”	150
\GlsXtrEnablePreLocationTag: new	25	long-noshort-sc-desc: renamed from “long-desc-sc”	150
\glstrfirstscfont: new	148	long-noshort-sm: renamed from “long-sm”	153
\glstrfirstsmfont: new	151	long-noshort-sm-desc: renamed from \long-desc-sm	154
\glstrlongshortdescsort: new	136	long-short-user: new	162
\glstrpostnamehook: added category check	104	long-short-user-desc: new	163
\glstrregularfont: new	27	\renewabbreviationstyle: new style: new	13
\glstruserfield: new	161		
\glstruserparen: new	161	1.05 (2016-06-10)	
\glstrusersuffix: new	162	\eglssetwidest: new	191
\GlsXtrWarnDeprecatedAbbrStyle: new	134	\glsFindWidestAnyName: new	193
short-em-long-em: new	157		
short-em-long-em-desc: new	158		
short-em-nolong: new	158		

\glsFindWidestAnyNameLocation: new	198	\@GLSfirst@: added check for nohyperfirst attribute	32
\glsFindWidestAnyNameSymbol: new	196	\@GLSfirstplural@: added check for nohyperfirst attribute	33
\glsFindWidestAnyNameSymbolLocation: new	197	\@GLSxtrp: new	52
\glsFindWidestLevelTwo: new	194	\@Glsfirst@: added check for nohyperfirst attribute	31
\glsFindWidestUsedAnyName: new	192	\@Glsfirstplural@: added check for nohyperfirst attribute	33
\glsFindWidestUsedAnyNameLocation: new	198	\@Glsxtrp: new	52
\glsFindWidestUsedAnyNameSymbol: new	195	\@gls@preglossaryhook: added \glossxtrsetpopts	111
\glsFindWidestUsedAnyNameSymbolLocation: new	196	\@glsfirst@: added check for nohyperfirst attribute	31
\glsFindWidestUsedLevelTwo: new	193	\@glsfirstplural@: added check for nohyperfirst attribute	32
\glsFindWidestUsedTopLevelName: new	192	\@glsxtrinmark: new	166
\glsfirstlongfootnotefont: new	139	\@glsxtrnotinmark: new	166
\glsgetwidestname: new	191	\@glsxtrp: new	51
\glsgetwidestsubname: new	192	\@glsxtrp@opt: new	51
\glslongfootnotefont: new	139	\glossxtrsetpopts: new	51
\glsxtrAltTreeIndent: new	191	\glsps: new	54
\glsxtraltttreeInit: new	191	\glspt: new	54
\glsxtrAltTreePar: new	191	\glsxtr@entry@p: new	52
\glsxtrAltTreeSetHangIndent: new	199	\glsxtrabbrvfootnote: new	139
\glsxtrAltTreeSetSubHangIndent: new	199	\glsxtrchecknohyperfirst: new	31
\glsxtraltttreeSubSymbolDescLocation: new	191	\glsxtrfieldtitlecasecs: new	98
\glsxtraltttreeSymbolDescLocation: new	190	\glsxtrifinmark: new	166
\glsxtrComputeTreeIndent: new	199	\GLSxtrp: new	55
\glsxtrComputeTreeSubIndent: new	199	\Glsxtrp: new	54
\glsxtrtreeopindent: new	191	\glsxtrp: new	53
short-em-long: fixed incorrect font used by long form	157	\glsxtrsetpopts: new	51
\xglsssetwidest: new	191	short-long-desc: added text key	138
1.06 (2016-06-18)		fixed misspelling of \glsabbrvfont in plural key	138
\@glsdoifexistssorwarn: new	9	long-short-desc: added missing text key	136
\@glsxtr@docdefval: new	8	fixed misspelling of \glsabbrvfont	137
\@glsxtr@useseesee: new	17	footnote: changed first forms to use \glsfirstlongfootnotefont	139
General: disabled docdef key at the start of the document	14	postfootnote: removed \footnote from first keys	141
docdef option changed to choice	8	switched from \glsfirstlongfont to \glsfirstlongfootnotefont	142
\glsxtr@useseesee: new	17	\RestoreAcronyms: modified	
\glsxtruseseesee: new	17	\@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse	71
\glsxtruseseeseeformat: new	17	1.08 (2016-12-13)	
\if@glsxtrdocdefrestricted: new	8	\@@glsxtr@record: new	6
1.07 (2016-08-15)			
\@@glsxtrp: new	51		

\@GLS@: added \@glsxtr@record	29	General: added record package option	7
\@GLSpl@: added \@glsxtr@record	29	\glsadd: added \@glsxtr@record	29
\@Gls@: added \@glsxtr@record	28	\glsdoifexists: now defines	
\@Glspl@: added \@glsxtr@record	29	\glslabel	15
\@gls@: added \@glsxtr@record	28	\glsxtr@@do@wrglossary: new	14
\@gls@@link@: added		\glsxtr@addloclistfield: new	7
\@glsxtr@record	29	\glsxtr@indexonly@saveentrycounter:	
\@gls@field@link: added		new	7
\@glsxtr@record	28	\glsxtr@record: new	6
\@gls@saveentrycounter: new	14	\glsxtr@resource: new	81
\@glsdispl: added \@glsxtr@record	29	\glsxtr@saveentrycounter: new	14
\@glspl@: added \@glsxtr@record	28	\glsxtr@setup@record: new	7
\@glsxtr@dorecord: new	6	\glsxtrassignfieldfont: added check	
\@glsxtr@err@undefaction: new	5	for existence	30
\@glsxtr@record: new	6	\glsxtrresourcefile: new	81
\@glsxtr@warn@onexistsordo: new	5	\printunsrtglossaries: new	81
\@glsxtr@warn@undefaction: new	5	\printunsrtglossary: new	81
\@print@unsrt@glossary: new	81		

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	
\.	112, 190
\@@cGLS@	59, 67
\@@cGLSpl@	59, 67
\@@cGLspl@	59, 67
\@@cgls@	59, 67
\@@cglspl@	59, 60, 67
\@@do@wrglossary	6
\@@do@wrglossary	7, 8, 14, 30
\@@glo@assign@sortkey	74
\@@glo@no@assign@sortkey	74
\@@glo@type	81
\@@glslocalreset	57
\@@glslocalunset	56
\@@glsreset	56
\@@glsunset	56
\@@glsxtr@autoindex@escspch	106–108
\@@glsxtr@checkspch	105, 106, 108
\@@glsxtr@disabledflycommand	23
\@@glsxtr@record	7, 8
\@@glsxtrp	51, 52
\@@glsxtrpostloctag	25
\@@glsxtrpreloctag	25, 26
\@@newglossaryentry@defcounters	58
\@@newglossaryentry@defunitcounters	65
\@@par	191
\@ACRlong	48
\@ACRlongpl	48
\@ACRshort	48
\@ACRshortpl	48
\@Acrlong	48
\@Acrlongpl	48
\@Acrshort	48
\@Acrshortpl	48
\@GLS@	29, 47, 61, 62
\@GLSdesc@	34
\@GLSpl@	47, 61, 62
\@GLSplural@	48
\@GLSsymbol@	35
\@GLStext@	48
\@GLSxtr@full	121
\@GLSxtr@fullpl	123
\@GLSxtr@p@acrlong@	48
\@GLSxtr@p@acrlongpl@	48
\@GLSxtr@p@acrshort@	48
\@GLSxtr@p@acrshortpl@	48
\@GLSxtr@p@long@	48
\@GLSxtr@p@longpl@	48
\@GLSxtr@p@plural@	47
\@GLSxtr@p@short@	48
\@GLSxtr@p@shortpl@	48
\@GLSxtr@p@text@	47
\@GLSxtrlong	48, 126
\@GLSxtrlongpl	48, 129
\@GLSxtrp	55, 56
\@GLSxtrshort	48, 124
\@GLSxtrshortpl	48, 127
\@Gls@	47, 61, 62
\@Gls@acentryname	69
\@Gls@entry@field	42, 54, 55
\@Gls@entryname	69
\@GlsXtrEnableOnTheFly	20
\@Glspl@	47, 61, 62
\@Glsplural@	48
\@Glstext@	48
\@Glsxtr	21, 22
\@Glsxtr@full	121
\@Glsxtr@fullpl	122
\@Glsxtr@p@acrlong@	48
\@Glsxtr@p@acrlongpl@	48
\@Glsxtr@p@acrshort@	48
\@Glsxtr@p@acrshortpl@	48
\@Glsxtr@p@long@	48
\@Glsxtr@p@longpl@	48

\@Glsxtr@p@plural@	47	\@glo@tmp	42
\@Glsxtr@p@short@	48	\@glo@type	18, 69, 72, 74, 76, 77, 79–81
\@Glsxtr@p@shortpl@	48	\@glo@types	95, 96, 192–198
\@Glsxtr@p@text@	47	\@glossary@default@style	23, 201
\@Glsxtr@long	48, 125	\@gls@	47, 60, 61
\@Glsxtr@longpl	48, 129	\@gls@@link	29
\@Glsxtrp	54, 55	\@gls@actualchar	105
\@Glsxtrpl	22	\@gls@adjustmode	29
\@Glsxtr@short	48, 124	\@gls@alt@hyp@opt	46
\@Glsxtr@shortpl	48, 127	\@gls@alt@hyp@opt@char	46
\@acrlong	48	\@gls@alt@hyp@opt@keys	46
\@acrlongpl	48	\@gls@automake	74
\@acrshort	48	\@gls@checkedmkidx	105, 106, 108
\@acrshortpl	48	\@gls@checkmkidxchars	105
\@alt@gls@hyp@opt	46	\@gls@codepage	77
\@auxout	6, 26, 59, 68, 72, 76, 77, 81	\@gls@counter	6, 30
\@cGLS	62	\@gls@currentlettergroup	81
\@cGLS@	59, 62, 67	\@gls@declareoption	4
\@cGLSpl	62	\@gls@doautomake	74
\@cGLSpl@	59, 62, 67	\@gls@encapchar	106
\@cGLspl@	59, 67	\@gls@entry@count	59
\@cgls@	59, 61, 67	\@gls@entry@field	42, 52–56, 58
\@cglspl@	59, 67	\@gls@entry@unitcount	67, 68
\@disable@onlypremakeg	73	\@gls@field@font	30–37
\@do@auxoutstuff	76, 77	\@gls@field@link	30–37, 42, 43
\@do@newglossaryentry	70, 118	\@gls@hyp@opt	42, 43, 46, 62, 120–129
\@empty	30, 38–41, 105, 106, 120–129	\@gls@hyp@opt@cs	46
\@end@glsxtr@addunused	18	\@gls@increment@currcount	58
\@end@glsxtr@usesee	17	\@gls@increment@currunitcount	66
\@endfortrue	132	\@gls@keymap	7, 17, 42
\@firstofone	30, 99, 104, 110	\@gls@label	6, 45, 46, 132
\@firstofthree		\@gls@levelchar	106
	30, 38–41, 46, 120, 122, 123, 125, 127, 128	\@gls@link	28, 29, 38–42, 120–129
\@firstoftwo	31–35, 39, 41, 44, 46,	\@gls@link@checkfirsthyper	71
	71, 113, 114, 121–123, 127–129, 166, 167	\@gls@link@nocheckfirsthyper	
\@for	13, 18, 57, 69, 72, 74, 81, 97, 109		28, 38–41, 120–129
\@glo@assign@sortkey	74	\@gls@local@increment@currcount	58
\@glo@category	63	\@gls@local@increment@currunitcount	66
\@glo@counterprefix	6	\@gls@loclist	75, 76
\@glo@countunit	63	\@gls@longpl	115–117
\@glo@default@sorttype	74	\@gls@noidx@do	81
\@glo@label	7, 17, 18, 42, 192–199	\@gls@noidx@nosanitizesort	74
\@glo@loclist	7	\@gls@noidx@sanitizesort	74
\@glo@name	105	\@gls@noidx@loclist@finalsep	75
\@glo@parent	193–195	\@gls@noidx@loclist@prev	75
\@glo@see	17, 18	\@gls@noidx@loclist@sep	75
\@glo@sort	105	\@gls@org@glsnoidx@displayloc	75, 76
\@glo@sorttype	74	\@gls@org@glsseeformat	75, 76
\@glo@thisvalue	161	\@gls@preglossaryhook	109

<code>\@gls@prevlevel</code>	199, 200	<code>\@glsxtr@csname</code>	64–67
<code>\@gls@quotechar</code>	105	<code>\@glsxtr@current@style</code>	23, 201
<code>\@gls@reference</code>	19, 72	<code>\@glsxtr@currentunitcount</code>	64–67
<code>\@gls@saveentrycounter</code>	7, 8, 14, 30	<code>\@glsxtr@currunitcount</code>	66, 68
<code>\@gls@setdefault@glslink@opts</code>	45	<code>\@glsxtr@declareoption</code>	4, 9, 10, 12
<code>\@gls@short</code>	117	<code>\@glsxtr@defaultnoglossarywarning</code> ..	12
<code>\@gls@shortpl</code>	115, 117	<code>\@glsxtr@deprecated@abbrstyle</code>	150, 151, 154, 155, 159–161
<code>\@gls@tmpb</code>	108	<code>\@glsxtr@disabledflycommand</code>	22
<code>\@gls@type</code>	74, 132, 192–198	<code>\@glsxtr@do@@wrindex</code>	45, 46
<code>\@gls@write@entrycounts</code>	59	<code>\@glsxtr@do@glsglssdisablehyperinlist</code> ..	44
<code>\@gls@write@entryunitcounts</code>	67	<code>\@glsxtr@do@style</code>	13, 183
<code>\@gls@write@entryunitcounts@do</code>	68	<code>\@glsxtr@do@titlecaps@warn</code> ..	99–102, 110
<code>\@glsabbrv@current@abbreviation</code> ..	116, 130	<code>\@glsxtr@doabbreviationsdef</code>	10
<code>\@glsacronymlists</code>	69	<code>\@glsxtr@doaccsupp</code>	12, 14
<code>\@glsdisp</code>	29	<code>\@glsxtr@docdefval</code>	8, 19
<code>\@glsdoifexistssorwarn</code>	8, 100–103	<code>\@glsxtr@doloctag</code>	25, 26
<code>\@glsentry</code>	59, 68	<code>\@glsxtr@dorecord</code>	6
<code>\@glslink</code>	47	<code>\@glsxtr@dostylewarn</code>	132
<code>\@glslocref</code>	6	<code>\@glsxtr@enabletagging</code>	109
<code>\@glsnumberformat</code>	6, 30, 104, 105	<code>\@glsxtr@end@</code>	20
<code>\@glsorder</code>	72	<code>\@glsxtr@endescspch</code>	105–108
<code>\@glspl@</code>	47, 60, 62	<code>\@glsxtr@entrycount@org@localreset</code> ..	58
<code>\@glsplural@</code>	48	<code>\@glsxtr@entrycount@org@localunset</code> ..	58
<code>\@glsplunc@token</code>	114	<code>\@glsxtr@entrycount@org@reset</code>	58
<code>\@glsstyle@alttree</code>	190	<code>\@glsxtr@entrycount@org@unset</code>	58
<code>\@glsstyle@inline</code>	190	<code>\@glsxtr@entryunitcount@org@localreset</code> ..	67
<code>\@glsstyle@listdotted</code>	185	<code>\@glsxtr@entryunitcount@org@localunset</code> ..	66
<code>\@glstarget</code>	47	<code>\@glsxtr@entryunitcount@org@reset</code> ..	66
<code>\@glstext@</code>	48	<code>\@glsxtr@entryunitcount@org@unset</code> ..	66
<code>\@glswidestname</code>	191, 192, 199	<code>\@glsxtr@err@undefaction</code>	5, 7
<code>\@glsxtr</code>	21, 22	<code>\@glsxtr@field@linkdefs</code>	28
<code>\@glsxtr@abbreviationsdef</code>	10, 14	<code>\@glsxtr@format@overridefalse</code>	104
<code>\@glsxtr@activate@initialtagging</code> ..	109, 111	<code>\@glsxtr@format@overridefalse</code>	104
<code>\@glsxtr@addunitcounter</code>	63	<code>\@glsxtr@foundinlist</code>	114
<code>\@glsxtr@addunusedxrefs</code>	18	<code>\@glsxtr@full</code>	120
<code>\@glsxtr@attrval</code>	98–103, 105	<code>\@glsxtr@fullpl</code>	122
<code>\@glsxtr@autoindex@at</code>	105, 106	<code>\@glsxtr@glossdescfont</code>	98–100
<code>\@glsxtr@autoindex@doextra@esc</code>	105	<code>\@glsxtr@glossnamefont</code>	100–103
<code>\@glsxtr@autoindex@encap</code>	105–107	<code>\@glsxtr@gobbleto@endescspch</code>	108
<code>\@glsxtr@autoindex@esc</code>	105, 107, 108	<code>\@glsxtr@idx@displaynumberlist</code>	73
<code>\@glsxtr@autoindex@escat</code>	105, 106	<code>\@glsxtr@idx@entrynumberlist</code>	73
<code>\@glsxtr@autoindex@escencap</code> ...	106, 107	<code>\@glsxtr@ifcsstart</code>	20
<code>\@glsxtr@autoindex@esclevel</code> ...	106, 107	<code>\@glsxtr@ifpunctoken</code>	114
<code>\@glsxtr@autoindex@escquote</code> ...	105, 107	<code>\@glsxtr@ifunitcounter</code>	63
<code>\@glsxtr@autoindex@level</code>	106, 107	<code>\@glsxtr@insert@dots</code>	116
<code>\@glsxtr@autoindex@setname</code>	105	<code>\@glsxtr@insert@dots@next</code>	116
<code>\@glsxtr@autoindexcrossrefs</code>	9, 17		
<code>\@glsxtr@cat</code>	57, 69, 109		

<code>\@glxtr@insertdots</code>	117	<code>\@glxtr@orgwarndep</code>	115
<code>\@glxtr@label</code>	18, 97, 98	<code>\@glxtr@p@acrlong@</code>	48
<code>\@glxtr@loadstyles</code>	184	<code>\@glxtr@p@acrlongpl@</code>	48
<code>\@glxtr@noidx@displaynumberlist</code>	73	<code>\@glxtr@p@acrshort@</code>	48
<code>\@glxtr@noidx@entrynumberlist</code>	73	<code>\@glxtr@p@acrshortpl@</code>	48
<code>\@glxtr@noidx@numberlistloop</code>	73	<code>\@glxtr@p@long@</code>	48
<code>\@glxtr@notfoundinlist</code>	114	<code>\@glxtr@p@longpl@</code>	48
<code>\@glxtr@optlist</code>	22	<code>\@glxtr@p@plural@</code>	47
<code>\@glxtr@org@GLS@</code>	29	<code>\@glxtr@p@short@</code>	47
<code>\@glxtr@org@GLSpl@</code>	29	<code>\@glxtr@p@shortpl@</code>	48
<code>\@glxtr@org@Gls@</code>	28	<code>\@glxtr@p@text@</code>	47
<code>\@glxtr@org@Glspl@</code>	29	<code>\@glxtr@pagetag</code>	25, 26
<code>\@glxtr@org@Glsxtrtitlefirst</code>	166, 167	<code>\@glxtr@pagetag</code>	25, 26
<code>\@glxtr@org@Glsxtrtitlefirstplural</code>	166, 167	<code>\@glxtr@prevunitcount</code>	66
<code>\@glxtr@org@Glsxtrtitlefull</code>	167, 168	<code>\@glxtr@record</code>	7, 8, 28, 29
<code>\@glxtr@org@Glsxtrtitlefullpl</code>	167, 168	<code>\@glxtr@redefstyles</code>	13, 183
<code>\@glxtr@org@Glsxtrtitlelong</code>	166, 168	<code>\@glxtr@reg@glosslist</code>	72–74
<code>\@glxtr@org@Glsxtrtitlelongpl</code>	166, 168	<code>\@glxtr@savepreloctag</code>	25, 26
<code>\@glxtr@org@Glsxtrtitleplural</code>	166, 167	<code>\@glxtr@setentrycountunsetattr</code>	57
<code>\@glxtr@org@Glsxtrtitleshort</code>	166, 167	<code>\@glxtr@setentryunitcountunsetattr</code>	69
<code>\@glxtr@org@Glsxtrtitleshortpl</code>	166, 167	<code>\@glxtr@setupshortcuts</code>	11, 12, 14
<code>\@glxtr@org@Glsxtrtitletext</code>	166, 167	<code>\@glxtr@swaptwo</code>	115
<code>\@glxtr@org@MakeUppercase</code>	166, 167	<code>\@glxtr@tag</code>	109
<code>\@glxtr@org@checkfirsthyper</code>	44, 71	<code>\@glxtr@taggingcs</code>	109
<code>\@glxtr@org@delimN</code>	26	<code>\@glxtr@thisloctag</code>	26
<code>\@glxtr@org@delimR</code>	26	<code>\@glxtr@tmp</code>	13
<code>\@glxtr@org@gls@</code>	28	<code>\@glxtr@type</code>	98
<code>\@glxtr@org@glsdisp</code>	29	<code>\@glxtr@unitcountlist</code>	63, 64
<code>\@glxtr@org@glsignore</code>	26	<code>\@glxtr@usesee</code>	17
<code>\@glxtr@org@glspl@</code>	28	<code>\@glxtr@warn@onexistsordo</code>	5, 7, 8
<code>\@glxtr@org@glsxtrtitlefirst</code>	166, 167	<code>\@glxtr@warn@undefaction</code>	5, 7, 8
<code>\@glxtr@org@glsxtrtitlefirstplural</code>	166, 167	<code>\@glxtr@docdeffalse</code>	19
<code>\@glxtr@org@glsxtrtitlefull</code>	166, 168	<code>\@glxtr@indexcrossrefsfalse</code>	9
<code>\@glxtr@org@glsxtrtitlefullpl</code>	166, 168	<code>\@glxtr@indexcrossrefstrue</code>	9
<code>\@glxtr@org@glsxtrtitlelong</code>	166, 167	<code>\@glxtrinmark</code>	165, 166
<code>\@glxtr@org@glsxtrtitlelongpl</code>	166, 167	<code>\@glxtrlong</code>	48, 125
<code>\@glxtr@org@glsxtrtitleplural</code>	166, 167	<code>\@glxtrlongpl</code>	48, 128
<code>\@glxtr@org@glsxtrtitleshort</code>	166, 167	<code>\@glxtrnotinmark</code>	165, 166
<code>\@glxtr@org@glsxtrtitleshortpl</code>	166, 167	<code>\@glxtrp</code>	53
<code>\@glxtr@org@glsxtrtitletext</code>	166, 167	<code>\@glxtrp@opt</code>	51
<code>\@glxtr@org@makeglossaries</code>	72	<code>\@glxtrpl</code>	21, 22
<code>\@glxtr@org@markboth</code>	165, 166	<code>\@glxtrpostloctag</code>	25, 27
<code>\@glxtr@org@markright</code>	165, 166	<code>\@glxtrpreloctag</code>	25, 27
<code>\@glxtr@org@newacronymstyle</code>	71	<code>\@glxtrshort</code>	47, 123
<code>\@glxtr@org@postdescription</code>	111	<code>\@glxtrshortpl</code>	48, 126
<code>\@glxtr@org@setacronymstyle</code>	70, 71	<code>\@glxtrundeftag</code>	5, 14
<code>\@glxtr@org@printglossary</code>	22	<code>\@gobble</code>	5, 7, 9, 30, 116
		<code>\@gobbletwo</code>	115
		<code>\@ifnextchar</code>	46

\@ifpackageloaded	70	\acrfullfmt	70
..... 4, 9, 82, 98, 100, 102, 104, 182		\ACRfullplfmt	70
\@ifstar	20, 46, 109	\Acrfullplfmt	70
\@ifundefined	182	\acrfullplfmt	70
\@input@	76	\acronymentry	70
\@istfilename	72	\acronymfont	38–41, 50, 71
\@makeglossary	72, 73	\acronymname	9
\@mfu@domakefirsttuc	110	\acronymsort	70
\@mfu@nocaplist	110	\acronymtype	10, 69–71
\@ne	59, 68	\acrpluralsuffix	70
\@newglossaryentry@defcounters ..	58, 65	\actualchar	107
\@newglossaryentryposthook	7, 42	\addtolength	200
\@newglossaryentryprehook	7, 42	\advance	59, 68
\@nnil	105, 106, 108, 114, 116	\AF	10
\@no@makeglossaries	80	\Af	10
\@onelevel@sanitize	22	\af	10
\@onlypreamble 23, 26, 68, 81, 104, 106, 107, 109		\AFP	10
\@print@unsrt@glossary	81	\Afp	10
\@printgloss@setsort	74	\afp	10
\@printglossary	22, 81	\AL	10
\@sGlsXtrEnableOnTheFly	20	\Al	10
\@secondofthree	31–	\al	10
33, 38–41, 43, 121, 122, 124, 125, 127, 129		\ALP	10
\@secondoftwo	30, 33–41, 44,	\Alp	10
47, 71, 115, 120, 121, 123–129, 141, 166, 167		\alp	10
\@thirdofthree	31–	\AnyTrackedLanguages	182
33, 38–41, 43, 121, 123, 124, 126, 128, 129		\appto 7, 13, 17, 42, 45, 46, 58, 65, 104, 114, 116	
\@thirdoftwo	33–37	\AS	10
\@warn@nomakeglossaries	77	\As	10
\@xdy@main@language	76	\as	10
\@xdy@language	76, 77	\ASP	10
		\Asp	10
_	78, 79	\asp	10
		\AtBeginDocument	14, 24
		\AtEndDocument	17, 59, 67, 76, 77
A			
\AB	10	B	
\Ab	10	babel package	105, 106, 113
\ab	10	\begin	61, 78, 79, 81, 185–189, 199
abbreviation styles:		\begingroup	6
long-noshort	159, 160		
short	144	C	
\abbreviationsname	9	category attributes:	
\abbrvpluralsuffix	117, 135,	discardperiod	113
137, 140, 142–144, 146, 148–155, 163, 164		entrycount	56, 57, 59, 68, 69
\ABP	10	firsttuc	102
\Abp	10	glossdesc	98
\abp	10	glossdescfont	98
\ACRfullfmt	70	glossname	100
\Acrfullfmt	70	glossnamefont	100, 102

<code>\firstacronymfont</code>	71, 72	<code>\GLS</code>	57, 69
<code>\footnote</code>	139	<code>\Gls</code>	21, 57, 69
<code>\forall glossaries</code> ...	18, 81, 96, 98, 192–198	<code>\gls</code>	21, 23, 57, 69, 78
<code>\forall glsentries</code>	59, 68	<code>\gls@assign@field</code>	7, 42
<code>\ForEachTrackedDialect</code>	182	<code>\gls@checkseeallowed</code>	19, 73
<code>\forall glsentries</code>	18, 96, 98, 192–198	<code>\gls@codepage</code>	77
<code>\forall listcsloop</code>	68	<code>\gls@defdocnewglossaryentry</code>	58, 65
<code>\forall listloop</code>	75, 110	<code>\gls@defglossaryentry</code>	21, 22
<code>\futurelet</code>	114	<code>\gls@save@numberlist</code>	24, 25, 27
G		<code>\gls@tmplen</code>	192–198, 200
<code>\gdef</code>	26, 106, 107	<code>\glsabbrvdefaultfont</code>	
<code>\Genacrfullformat</code>	70	119, 136, 137, 140, 142–144, 146
<code>\genacrfullformat</code>	70	<code>\glsabbrvmfont</code>	155–161
<code>\GenericAcronymFields</code>	70	<code>\glsabbrvfont</code>	49, 71, 119, 123,
<code>\Genplacrfullformat</code>	70	124, 127, 128, 130, 131, 135–144, 146–164
<code>\genplacrfullformat</code>	70	<code>\glsabbrvuserfont</code>	162–164
<code>\glo@name</code>	101, 102	<code>\glsabrvfont</code>	
<code>glossaries</code> package	184	135, 137, 139, 141, 156, 157, 162, 164
<code>glossaries-accsupp</code> package	12, 14, 82	<code>\GLSaccessdesc</code>	34
<code>glossaries-extra</code> package	2	<code>\Glsaccessdesc</code>	34, 99, 108
<code>glossaries-extra-stylemods</code> package .	12, 111, 183	<code>\glsaccessdesc</code>	34, 99, 112
<code>\GlossariesExtraWarning</code>	5, 9,	<code>\GLSaccessdescplural</code>	34
.....	20, 22, 71, 78, 81, 98–101, 103, 109, 110, 134	<code>\Glsaccessdescplural</code>	34
<code>\GlossariesExtraWarningNoLine</code> .	9, 59, 68	<code>\glsaccessdescplural</code>	34
<code>\GlossariesWarning</code>	25, 75, 76, 132	<code>\GLSaccessfirst</code>	32
<code>\GlossariesWarningNoLine</code>	73, 77	<code>\Glsaccessfirst</code>	31
<code>glossary</code> styles:		<code>\glsaccessfirst</code>	31
<code>alttree</code>	190, 191, 199	<code>\GLSaccessfirstplural</code>	33
<code>inline</code>	190	<code>\Glsaccessfirstplural</code>	33
<code>listdotted</code>	185	<code>\glsaccessfirstplural</code>	32
<code>listdottedstyle</code>	185	<code>\Glsaccesslong</code> 40, 118, 126, 136, 143, 146, 163	
<code>sublistdotted</code>	185	<code>\glsaccesslong</code>	40, 118, 125, 126,
<code>glossary-long</code> package	186	136, 138, 140, 142, 143, 145–147, 163, 164
<code>glossary-longbooktabs</code> package	186	<code>\Glsaccesslongpl</code>	
<code>glossary-mcols</code> package	184	41, 119, 129, 136, 143, 146, 163
<code>glossary-tree</code> package	184	<code>\glsaccesslongpl</code>	41, 118, 128,
<code>\glossaryentrynumbers</code>	27	129, 136, 138, 140–143, 145–147, 163, 164
<code>\glossaryheader</code>	81, 185–189, 200	<code>\GLSaccessname</code>	33
<code>\glossarypostamble</code>	81	<code>\Glsaccessname</code>	33
<code>\glossarypreamble</code>	81	<code>\glsaccessname</code>	33
<code>\glossarysection</code>	79, 81	<code>\GLSaccessplural</code>	32
<code>\glossarytitle</code>	79, 81	<code>\Glsaccessplural</code>	32
<code>\glossarytoctitle</code>	79, 81	<code>\glsaccessplural</code>	32
<code>\glossentry</code>	185–189, 200	<code>\Glsaccessshort</code>	
<code>\glossentrydesc</code>	185–191	38, 124, 130, 138, 140, 142, 145, 164
<code>\glossentryname</code>	185–189, 200	<code>\glsaccessshort</code>	38, 118, 123, 124,
<code>\glossentrysymbol</code>	186–191	130, 131, 136, 138, 140, 142–146, 163, 164
<code>\glossxtrsetpopts</code>	111	<code>\Glsaccessshortpl</code>	
		39, 127, 130, 138, 140–142, 145, 164

<code>\glsaccessshortpl</code>	39, 118, 119, 127, 128, 130, 136, 138, 140, 142–146, 163, 164
<code>\GLSaccesssymbol</code>	35
<code>\Glsaccesssymbol</code>	35, 109
<code>\glsaccesssymbol</code>	35, 108, 112
<code>\GLSaccesssymbolplural</code>	35
<code>\Glsaccesssymbolplural</code>	35
<code>\glsaccesssymbolplural</code>	35
<code>\GLSaccessstext</code>	31
<code>\Glsaccessstext</code>	31
<code>\glsaccessstext</code>	30
<code>\glsacrshortcutstrue</code>	11, 12
<code>\glsacspacemax</code>	72
<code>\glsadd</code>	18, 78
<code>\glsaddstoragekey</code>	92
<code>\glsbackslash</code>	20
<code>\glscapscase</code>	30–41, 43, 120–131
<code>\glscategory</code> ...	27, 30, 44, 49, 93–95, 98– 104, 108, 109, 111, 112, 120–124, 126–128
<code>\glscategorylabel</code>	44, 115–117, 141
<code>\glsclosebrace</code>	79, 80
<code>\glscurrententrylabel</code>	25, 26, 81, 104, 110, 111
<code>\glscurrentfieldvalue</code>	161
<code>\glscustomtext</code>	28, 38–41, 120–130, 132
<code>\glsdefaulttype</code>	9, 78, 81
<code>\glsdescriptionaccessdisplay</code>	86, 99
<code>\glsdescriptionpluralaccessdisplay</code>	86, 87
<code>\glsdescwidth</code>	185, 187–189
<code>\glsdetoklabel</code>	15–20, 30, 58, 59, 64–68, 75, 76, 98, 101, 102, 193–195
<code>\glsdisplaynumberlist</code>	73, 75
<code>\glsdohyperlink</code>	47
<code>\glsdoifexists</code>	8, 17, 28, 29, 38–41, 75, 76, 120–129
<code>\glsdoifexistsordo</code>	29
<code>\glsdoifexistsorwarn</code> ...	9, 98, 99, 108, 109
<code>\glsdohyperlink</code>	47
<code>\glsdosanitizesort</code>	74
<code>\glsenableentrycount</code>	57, 59, 67
<code>\glsenableentryunitcount</code>	59, 68
<code>\glsentrycurrcount</code>	58, 59, 66
<code>\Glsentrydesc</code>	86, 91, 99
<code>\glsentrydesc</code>	86, 91, 100
<code>\Glsentrydescplural</code>	87, 91
<code>\glsentrydescplural</code>	86, 87, 91
<code>\Glsentryfirst</code>	63, 84, 90
<code>\glsentryfirst</code>	63, 84, 90, 179
<code>\Glsentryfirstplural</code>	63, 84, 90
<code>\glsentryfirstplural</code> ...	63, 84, 85, 90, 179
<code>\Glsentryfull</code>	70
<code>\glsentryfull</code>	70
<code>\Glsentryfullpl</code>	70
<code>\glsentryfullpl</code>	70
<code>\glsentryitem</code>	185–189, 200
<code>\Glsentrylong</code>	50, 63, 88, 92
<code>\glsentrylong</code>	49, 50, 63, 88, 92, 141, 180
<code>\Glsentrylongpl</code>	50, 51, 63, 89, 92
<code>\glsentrylongpl</code> ..	50, 51, 63, 89, 92, 180, 181
<code>\Glsentryname</code>	82, 89, 100, 102, 103
<code>\glsentryname</code>	82, 89, 105, 192–199
<code>\glsentrynumberlist</code>	73, 76, 197–199
<code>\Glsentryplural</code>	83, 90
<code>\glsentryplural</code>	83, 90, 178
<code>\glsentryprevcount</code>	58, 60, 66
<code>\glsentryprevmaxcount</code>	66
<code>\glsentryprevtotalcount</code>	66
<code>\Glsentryshort</code>	49, 50, 87, 91
<code>\glsentryshort</code> ...	49, 50, 72, 87, 91, 176, 177
<code>\Glsentryshortpl</code>	49, 50, 88, 92
<code>\glsentryshortpl</code> ..	49, 50, 88, 91, 92, 176, 177
<code>\Glsentrysymbol</code>	85, 90
<code>\glsentrysymbol</code>	85, 90, 196, 197
<code>\Glsentrysymbolplural</code>	86, 91
<code>\glsentrysymbolplural</code>	85, 86, 91
<code>\Glsentrytext</code>	83, 89
<code>\glsentrytext</code>	82, 83, 89, 90, 177, 178
<code>\Glsentryuseri</code>	36
<code>\glsentryuseri</code>	36
<code>\Glsentryuserii</code>	36
<code>\glsentryuserii</code>	36
<code>\Glsentryuseriii</code>	36
<code>\glsentryuseriii</code>	36
<code>\Glsentryuseriv</code>	37
<code>\glsentryuseriv</code>	37
<code>\Glsentryuserv</code>	37
<code>\glsentryuserv</code>	37
<code>\Glsentryuservi</code>	37
<code>\glsentryuservi</code>	37
<code>\glsfieldxdef</code>	97, 98
<code>\glsfindwidesttoplevelname</code>	192
<code>\GLSfirst</code>	171, 172
<code>\Glsfirst</code>	172
<code>\glsfirst</code>	171
<code>\glsfirstabbrvdefaultfont</code>	119, 136, 137, 140, 142–144, 146
<code>\glsfirstabbrvmfont</code>	156–161

<code>\glsfirstabbrvfont</code>	<code>\glslinkcheckfirsthyperhook</code>
..... 71, 118, 119, 135–146, 148–164	44
<code>\glsfirstabbrvuserfont</code>	<code>\glslinkvar</code>
163, 164	46
<code>\glsfirstaccessdisplay</code>	<code>\glslistdottedwidth</code>
84	185
<code>\glsfirstlongdefaultfont</code>	<code>\glslongaccessdisplay</code>
..... 136, 137, 143, 145, 146	88
<code>\glsfirstlongemfont</code>	<code>\glslongdefaultfont</code>
156–158, 160 120, 136, 137, 139, 143, 145, 146
<code>\glsfirstlongfont</code> ... 118, 119, 135–138,	<code>\glslongemfont</code>
140, 142, 143, 145–147, 156–160, 162–164	155–160
<code>\glsfirstlongfootnotefont</code>	<code>\glslongfont</code>
139–142	50,
<code>\glsfirstlonguserfont</code>	120, 125, 126, 128, 129, 136, 137, 140,
163, 164	142, 143, 145, 146, 156–158, 160, 163, 164
<code>\GLSfirstplural</code>	<code>\glslongfootnotefont</code>
172	139, 140, 142
<code>\Glsfirstplural</code>	<code>\glslongpltok</code>
172, 173	117, 118,
<code>\glsfirstplural</code>	135–139, 146, 147, 156, 157, 159, 162, 164
172	<code>\glslongpluralaccessdisplay</code>
<code>\glsfirstpluralaccessdisplay</code>	89
84, 85	<code>\glslongtok</code> .. 70, 116, 118, 135–139, 141,
<code>\glsforeachincategory</code>	143, 144, 146, 147, 156, 157, 159, 162, 163
132	<code>\glslonguserfont</code>
<code>\glsgenentryfmt</code>	162–164
27	<code>\glsnameaccessdisplay</code> ... 82, 100, 101, 103
<code>\glsgetattribute</code> 60, 64–66, 98–101, 103, 105	<code>\glsnamefont</code>
<code>\glsgetcategoryattribute</code>	100, 102, 103
93	<code>\glsnoidxdisplayloc</code>
<code>\glsgetwidestname</code>	75, 76
191	<code>\glsnoidxdisplayloclisthandler</code>
<code>\glsgroupheading</code>	75
185–189, 200	<code>\glsnoidxloclist</code>
<code>\glsgroupskip</code>	76
186–190, 201	<code>\glsnoidxnumberlistloophandler</code>
<code>\glshasattribute</code> 59, 60, 64–68, 98–	75
104, 135, 137–139, 141, 156, 158, 162, 164	<code>\glsnonnumberlistfalse</code>
<code>\glshascategoryattribute</code>	25
94	<code>\glsnonnumberlisttrue</code>
<code>\glshypernumber</code>	25
104	<code>\glsnumberlistloop</code>
<code>\glsifattribute</code>	73
31,	<code>\glsnumlistlastsep</code>
44, 45, 52, 96, 99–102, 110, 113, 168–176	75
<code>\glsifcategory</code>	<code>\glsnumlistsep</code>
96	75
<code>\glsifcategoryattribute</code> ... 44, 94, 95, 117	<code>\glsopenbrace</code>
<code>\glsifnotregular</code>	79, 80
30	<code>\glsorder</code>
<code>\glsifnotregularcategory</code>	72
95	<code>\glspagelistwidth</code>
<code>\glsifplural</code> . 30, 32–35, 38–41, 113, 120–131	185, 187–189
<code>\glsifregular</code>	<code>\GLSpl</code>
27, 30, 63	57, 69
<code>\glsifregularcategory</code>	<code>\Glspl</code>
95	22, 57, 69
<code>\glsignore</code>	<code>\glspl</code>
26	22, 57, 69
<code>\glsinlinedescformat</code>	<code>\GLSplural</code>
190	170, 171
<code>\glsinlinesubdescformat</code>	<code>\Glsplural</code>
190	171
<code>\glsinsert</code>	<code>\glsplural</code>
30, 38–41, 120–131	170, 171
<code>\glskeylisttok</code>	<code>\glspluralaccessdisplay</code>
70, 116, 118	83
<code>\glslabel</code> ... 15, 27, 44, 71, 112, 130, 131, 141	<code>\glspluralsuffix</code>
<code>\glslabeltok</code> 70, 116, 118, 135–139,	116, 120,
141, 143, 144, 146, 147, 156–159, 162–164	135, 137, 140, 142–144, 146, 148, 152, 162
<code>\glsletentryfield</code>	<code>\glspostdescription</code>
105	111, 185–191
<code>\glslink</code>	<code>\glspostinline</code>
70	190
<code>\glslink options</code>	<code>\glspostlinkhook</code> . 28, 29, 38–42, 51, 120–129
format	<code>\glsprestandardsort</code>
104	73
hyper	<code>\glsresetentrylist</code>
165	81
noindex	<code>\glsseeformat</code>
6, 44, 165	17, 75, 76
	<code>\glssetabbrvfmt</code>
	27,
	30, 49, 98–103, 108, 109, 120–124, 126–128

<code>\glsssetattribute</code>	135, 137–139, 141, 143, 144, 146, 147, 156, 158, 159, 162, 164
<code>\glsssetcategoryattribute</code>	57, 69, 71, 93, 94, 96, 97, 109
<code>\glssshortaccessdisplay</code>	87
<code>\glssshortpltok</code>	117, 118, 135–139, 141, 143, 144, 156, 157, 162, 164
<code>\glssshortpluralaccessdisplay</code>	88
<code>\glssshorttok</code>	70, 116–118, 135–139, 141, 143, 144, 147, 156, 157, 159, 162, 163
<code>\glssubentryitem</code>	185–190, 200
<code>\glssymbolaccessdisplay</code>	85
<code>\glssymbolpluralaccessdisplay</code>	85, 86
<code>\glstarget</code>	185–190, 200
<code>\GLstext</code>	170
<code>\Glstext</code>	170
<code>\glstext</code>	170
<code>\glstextaccessdisplay</code>	82, 83
<code>\glstextformat</code>	29
<code>\glstextup</code>	148
<code>\glstreeindent</code>	199, 200
<code>\glstreenamebox</code>	200
<code>\glstreenamefmt</code>	191–200
<code>\glstype</code>	38–42, 120–129
<code>\glunset</code>	18, 60, 61
<code>\glswrite</code>	72
<code>\glswriteentry</code>	6
<code>\Glsxtr</code>	22
<code>\glxtr</code>	22
<code>\glxtr@do@wrglossary</code>	6–8
<code>\glxtr@addlocloclistfield</code>	8
<code>\glxtr@addunused</code>	18
<code>\glxtr@applyabbrvfmt</code>	130
<code>\glxtr@applyabbrvstyle</code>	115, 116, 132
<code>\glxtr@doooption</code>	4, 9, 13, 14
<code>\glxtr@headentry@p</code>	52, 53
<code>\glxtr@ifnextpunc</code>	114
<code>\glxtr@ifpunctoken</code>	114
<code>\glxtr@indexonly@saveentrycounter</code>	7, 8, 14
<code>\glxtr@keylist</code>	21, 22
<code>\glxtr@next</code>	114
<code>\glxtr@orgmakenoidxglossaries</code>	18, 19
<code>\glxtr@puncloclist</code>	114
<code>\glxtr@record</code>	6
<code>\glxtr@resource</code>	81
<code>\glxtr@saveentrycounter</code>	6, 7
<code>\glxtr@setup@record</code>	7, 8, 14
<code>\glxtr@usesee</code>	17
<code>\glxtr@warnonexistsordo</code>	5, 7, 8, 16
<code>\glxtrabbrvfootnote</code>	139–141
<code>\glxtrabbrvtype</code>	9, 10, 118
<code>\glxtraddallcrossrefs</code>	17
<code>\glxtrAltTreeIndent</code>	191
<code>\glxtralttreeInit</code>	199
<code>\glxtrAltTreePar</code>	190
<code>\glxtrAltTreeSetHangIndent</code>	191, 200
<code>\glxtrAltTreeSetSubHangIndent</code>	200
<code>\glxtralttreeSubSymbolDescLocation</code>	200
<code>\glxtralttreeSymbolDescLocation</code>	191, 200
<code>\glxtrassignfieldfont</code>	30–37
<code>\glxtrcat</code>	21, 22
<code>\glxtrchecknohyperfirst</code>	31–33
<code>\glxtrComputeTreeIndent</code>	200
<code>\glxtrComputeTreeSubIndent</code>	200
<code>\GlsXtrDefineAbbreviationShortcuts</code>	11, 12
<code>\GlsXtrDefineOtherShortcuts</code>	11, 12
<code>\glxtrdiscardperiod</code>	112
<code>\glxtrdoautoindexname</code>	45, 46, 104
<code>\glxtrdopostpunc</code>	141
<code>\glxtrdowrglossaryhook</code>	45, 46
<code>\GlsXtrEnableEntryCounting</code>	69
<code>\GlsXtrEnableEntryUnitCounting</code>	57
<code>\GlsXtrEnableOnTheFly</code>	20, 23
<code>\glxtrfieldtitlecase</code>	99–102
<code>\glxtrfieldtitlecasesecs</code>	98
<code>\glxtrfirstscfont</code>	148–151
<code>\glxtrfirstsmfont</code>	152–155
<code>\GlsXtrFormatLocationList</code>	25, 27, 197–199
<code>\GLSxtrfull</code>	10, 174, 175
<code>\Glsxtrfull</code>	10, 175, 176
<code>\glxtrfull</code>	10, 175
<code>\Glsxtrfullformat</code>	119, 131, 133, 136, 138, 140, 142, 144, 145, 147, 163, 164
<code>\glxtrfullformat</code>	119, 131, 133, 136–138, 140, 142, 144–146, 163, 164
<code>\GLSxtrfullpl</code>	10, 175, 176
<code>\Glsxtrfullpl</code>	10, 176
<code>\glxtrfullpl</code>	10, 175
<code>\Glsxtrfullplformat</code>	119, 131, 133, 136, 138, 140, 142, 144, 145, 147, 163, 164
<code>\glxtrfullplformat</code>	131, 133, 136, 138, 140, 142, 144, 145, 147, 163, 164
<code>\glxtrfullsep</code>	118, 119, 135–138, 140–143, 145, 146, 156, 157, 161
<code>\glxtrgenabbrvfmt</code>	27

<code>\Glsxtrheadfirst</code>	167	<code>\GlsXtrNoGlsWarningAutoMake</code>	80
<code>\glsxtrheadfirst</code>	167	<code>\GlsXtrNoGlsWarningBuildInfo</code>	80
<code>\Glsxtrheadfirstplural</code>	167	<code>\GlsXtrNoGlsWarningCheckFile</code>	80
<code>\glsxtrheadfirstplural</code>	167	<code>\GlsXtrNoGlsWarningEmptyMain</code>	80
<code>\Glsxtrheadfull</code>	167	<code>\GlsXtrNoGlsWarningEmptyNotMain</code>	80
<code>\glsxtrheadfull</code>	167	<code>\GlsXtrNoGlsWarningEmptyStart</code>	80
<code>\Glsxtrheadfullpl</code>	167	<code>\GlsXtrNoGlsWarningHead</code>	79
<code>\glsxtrheadfullpl</code>	167	<code>\GlsXtrNoGlsWarningMisMatch</code>	80
<code>\Glsxtrheadlong</code>	167	<code>\GlsXtrNoGlsWarningNoOut</code>	80
<code>\glsxtrheadlong</code>	167	<code>\GlsXtrNoGlsWarningTail</code>	80
<code>\Glsxtrheadlongpl</code>	167	<code>\glsxtrorg@ifKV@glslink@hyper</code>	28
<code>\glsxtrheadlongpl</code>	167	<code>\GLSxtrp</code>	52
<code>\Glsxtrheadplural</code>	167	<code>\Glsxtrp</code>	52
<code>\glsxtrheadplural</code>	167	<code>\glsxtrp</code>	51, 54
<code>\Glsxtrheadshort</code>	167	<code>\Glsxtrpl</code>	22
<code>\glsxtrheadshort</code>	167	<code>\glsxtrpl</code>	22
<code>\Glsxtrheadshortpl</code>	167	<code>\glsxtrpostdescription</code>	96, 111, 190
<code>\glsxtrheadshortpl</code>	167	<code>\glsxtrpostlink</code>	112
<code>\Glsxtrheadtext</code>	167	<code>\glsxtrpostlinkendsentence</code>	112
<code>\glsxtrheadtext</code>	167	<code>\glsxtrpostlinkhook</code>	112
<code>\glsxtrifcounttrigger</code>	60, 61	<code>\glsxtrpostlocalreset</code>	57, 58, 67
<code>\glsxtrifemptyglossary</code>	79	<code>\glsxtrpostlocalunset</code>	56, 58, 66
<code>\glsxtrifindexing</code>	45	<code>\glsxtrpostnamehook</code>	101–103
<code>\glsxtrifinmark</code>	53–55, 166, 167	<code>\GlsXtrPostNewAbbreviation</code>	
<code>\glsxtrifnextpunc</code>	114, 115	118, 133, 135, 137–139, 141,
<code>\glsxtrifperiod</code>	113	143, 144, 146, 147, 156, 158, 159, 162, 164
<code>\glsxtrifwasfirstuse</code>	30–33,	<code>\glsxtrpostreset</code>	56, 58, 66
38–41, 44, 71, 112, 113, 121, 123–129, 141		<code>\glsxtrpostunset</code>	56, 58, 66
<code>\Glsxtrinlinefullformat</code>		<code>\glsxtrprotectlinks</code>	47
119, 121, 133, 140, 142, 143, 145, 146, 181		<code>\glsxtrregularfont</code>	27, 30
<code>\glsxtrinlinefullformat</code>		<code>\glsxtrrestoremarkhook</code>	165, 166
.....	119–121, 133, 140, 142–146, 181	<code>\glsxtrscfont</code>	148–151
<code>\Glsxtrinlinefullplformat</code>		<code>\glsxtrscsuffix</code>	148–151
119, 122, 133, 140, 142, 143, 145, 146, 182		<code>\GlsXtrSetActualChar</code>	107
<code>\glsxtrinlinefullplformat</code>	119,	<code>\GlsXtrSetEncapChar</code>	108
122, 123, 133, 140, 142, 143, 145, 146, 181		<code>\GlsXtrSetEscChar</code>	107
<code>\glsxtrinsetinsidefalse</code>	135	<code>\GlsXtrSetLevelChar</code>	107
<code>\GLSxtrlong</code>	10, 173, 174	<code>\glsxtrsetpopts</code>	51
<code>\Glsxtrlong</code>	10, 174	<code>\glsxtrsetupfulldefs</code>	120–123, 141
<code>\glsxtrlong</code>	10, 173	<code>\GLSxtrshort</code>	10, 55, 56, 168, 169
<code>\GLSxtrlongpl</code>	10, 173, 174	<code>\Glsxtrshort</code>	10, 169
<code>\Glsxtrlongpl</code>	10, 174	<code>\glsxtrshort</code>	10, 168
<code>\glsxtrlongpl</code>	10, 173	<code>\GLSxtrshortpl</code>	10, 168, 169
<code>\glsxtrlongshortdescsort</code>	136	<code>\Glsxtrshortpl</code>	10, 169
<code>\glsxtrmarkhook</code>	165	<code>\glsxtrshortpl</code>	10, 168, 169
<code>\glsxtrnewabbrevpresetkeyhook</code>	117	<code>\glsxtrsmfont</code>	151–155
<code>\glsxtrnewnumber</code>	11	<code>\glsxtrsmsuffix</code>	152–155
<code>\glsxtrnewsymbol</code>	11	<code>\glsxtrtagfont</code>	110
<code>\glsxtrNoGlossaryWarning</code>	12, 76	<code>\Glsxtrtitlefirst</code>	166, 167, 179

<code>\glxstrtitlefirst</code>	166, 167, 179	<code>\ifcsdef</code>	6, 42, 43, 51–55, 64, 98–101, 103, 112, 115, 130, 133, 185–189
<code>\Glsxstrtitlefirstplural</code>	166, 167, 179, 180	<code>\ifcsempy</code>	81
<code>\glxstrtitlefirstplural</code>	166, 167, 179	<code>\ifcsstring</code>	15, 94, 132
<code>\Glsxstrtitlefull</code>	167, 168, 181	<code>\ifcsundef</code>	19, 23, 25, 47, 58, 64–67, 76, 94, 132–134, 184, 192, 199
<code>\glxstrtitlefull</code>	166–168, 181	<code>\ifcsvoid</code>	93
<code>\Glsxstrtitlefullpl</code>	167, 168, 182	<code>\ifdef</code>	11, 16, 23, 24, 44, 53–55, 75, 76, 96, 97, 107, 108, 111, 161, 176–182, 185, 190
<code>\glxstrtitlefullpl</code>	166–168, 181, 182	<code>\ifdefempty</code> ...	17, 57, 69, 72, 74, 81, 109, 130
<code>\Glsxstrtitlelong</code>	166–168, 180	<code>\ifdefequal</code>	80
<code>\glxstrtitlelong</code>	166, 167, 180	<code>\ifdefstring</code>	105, 110
<code>\Glsxstrtitlelongpl</code>	166–168, 181	<code>\ifdefvoid</code>	17, 18, 63
<code>\glxstrtitlelongpl</code>	166, 167, 180	<code>\ifdim</code>	24, 72, 192–199
<code>\Glsxstrtitleplural</code>	166, 167, 178	<code>\IfFileExists</code>	13, 76, 80, 184
<code>\glxstrtitleplural</code>	166, 167, 178	<code>\ifglossaryexists</code>	15, 16
<code>\Glsxstrtitleshort</code>	166, 167, 177	<code>\ifglsacronym</code>	10, 80
<code>\glxstrtitleshort</code>	166, 167, 176	<code>\ifglsautomake</code>	74, 80
<code>\Glsxstrtitleshortpl</code>	166, 167, 177	<code>\ifglsentryexists</code> 15, 16, 21, 22, 25, 30, 94, 111, 112
<code>\glxstrtitleshortpl</code>	166, 167, 176, 177	<code>\ifglsfieldeq</code>	92
<code>\Glsxstrtitletext</code>	166, 167, 178	<code>\ifglsfield</code>	161
<code>\glxstrtitletext</code>	166, 167, 177, 178	<code>\ifglschaslong</code>	63
<code>\glxstrtreetopindent</code>	191, 199	<code>\ifglschasparent</code>	192–195
<code>\glxstrundefaction</code>	5, 7, 8, 15, 16	<code>\ifglschasshort</code>	27, 30
<code>\glxstrundeftag</code>	14	<code>\ifglschassymbol</code>	112, 191
<code>\GlsXtrUseAbbrStyleFmts</code> 137, 139, 147–161, 163, 165	<code>\ifglsindexonlyfirst</code>	45
<code>\GlsXtrUseAbbrStyleSetup</code>	148–161, 163, 164	<code>\ifglsnogroupskip</code>	186–190, 201
<code>\glxstruserfield</code>	161	<code>\ifglsnonumberlist</code>	27
<code>\glxstruserparen</code>	162–164	<code>\ifglsanitizesort</code>	74
<code>\glxstrusersuffix</code>	163, 164	<code>\ifglsused</code>	18, 44, 45, 59, 68, 71, 130, 192, 193, 195, 197, 198
<code>\glxstruseseeformat</code>	17	<code>\ifglxindy</code>	76, 78
<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	115, 134	<code>\ifglxtrininsertinside</code> 123–129, 136, 138, 140–147, 163, 164
<code>\GlsXtrWarning</code>	21, 22	<code>\ifHy@hyperindex</code>	104
H			
<code>\hangindent</code>	191, 199	<code>\ifinlistcs</code>	19
<code>\hbox</code>	185	<code>\ifKV@glslink@hyper</code>	28
<code>\hfill</code>	185	<code>\ifKV@glslink@noindex</code>	6, 7, 45
<code>\hsize</code>	24	<code>\ifnum</code>	8, 59, 60, 67, 68, 200
<code>\hss</code>	185	<code>\ifthenelse</code>	80
<code>\hyperlink</code>	47	<code>\IfTrackedLanguageFileExists</code>	182
<code>\hyperpage</code>	104	<code>\ifundef</code>	47, 72, 109, 110
<code>hyperref</code> package	47, 104, 165, 176	<code>\ifx</code>	23, 25, 105, 106, 108, 114, 116, 201
I			
<code>\if</code>	20	<code>\immediate</code>	59, 68, 76, 77
<code>\if@glxstr@format@override</code>	105	<code>\index</code>	105
<code>\if@glxstrdocdefrestricted</code>	19	<code>\indexspace</code>	201
<code>\if@glxstrindexcrossrefs</code>	9, 17	<code>\input</code>	182
<code>\ifblank</code>	13, 21, 22, 72	<code>\InputIfFileExists</code>	81
<code>\ifcase</code>	5, 7, 11, 12, 19		

<code>\istfilename</code>	72	<code>mfirstuc package</code>	109, 110
<code>\item</code>	78, 79, 185	<code>\mfirstucMakeUppercase</code>	
		..	31–41, 43, 49–52, 55, 56, 62, 70, 82–
			92, 101, 102, 121, 123, 124, 126, 128–131
J			
<code>\jobname</code>	76, 78–80	<code>\mfu@checkword@arg</code>	110
		<code>\mfu@checkword@do</code>	110
K			
<code>\key@ifundefined</code>	7, 42	N	
<code>\KV@glslink@hyperfalse</code>	31, 44, 47	<code>\NeedsTeXFormat</code>	4, 184
<code>\KV@glslink@noindexfalse</code>	44, 45	<code>\new@glossaryentry</code>	19, 74
<code>\KV@glslink@noindextrue</code>	47	<code>\new@ifnextchar</code>	42, 43, 62, 113, 120–129
L			
<code>\LaTeX</code>	78, 79	<code>\newabbr</code>	10
<code>\leaders</code>	185	<code>\newabbreviation</code>	10, 71
<code>\let</code>	4–11, 14, 18, 19, 22, 23, 25, 26, 28–	<code>\newabbreviationhook</code>	118
	41, 43, 44, 46–48, 51, 57–59, 66, 67, 69–	<code>\newabbreviationstyle</code>	
	73, 75, 76, 99, 100, 102–106, 109–111,	135–139, 141, 143, 144, 146–164
	114–117, 120–129, 141, 165–168, 190, 192	<code>\newacronym</code>	69, 71
<code>\letabbreviationstyle</code>	141,	<code>\newacronymhook</code>	70
	142, 144, 145, 147, 149, 150, 153, 158, 159	<code>\newacronymstyle</code>	71
<code>\letcs</code>	17, 18, 42, 75, 76, 98–103	<code>\newcommand</code> 4–18, 20–23, 25–27, 30, 31, 42,	
<code>\levelchar</code>	107	43, 45–47, 51, 53–58, 60, 62–66, 68, 69,	
<code>\listadd</code>	63	71, 72, 75–79, 81–98, 104–116, 118–130,	
<code>\listbreak</code>	110	132–134, 136, 139, 147, 148, 151, 152,	
<code>\listcseadd</code>	65	155, 161, 162, 166–182, 184, 190–192, 199	
<code>\listcsgadd</code>	19	<code>\newcount</code>	8
<code>\listcsxadd</code>	64	<code>\newentry</code>	11
<code>\loadglsentries</code>	19, 78	<code>\newglossary</code>	9, 72, 73
		<code>\newglossaryentry</code> 11, 19, 58, 65, 70, 96, 97, 118	
M			
<code>\MakeAcronymsAbbreviations</code>	71	<code>\newglossaryentry options</code>	
<code>\makeatletter</code>	76, 106	desc	86, 91
<code>\makeatother</code>	106	descplural	86, 87, 91
<code>\makebox</code>	185, 200	first ...	47, 83, 84, 90, 135, 171–173, 178, 202
<code>\makefirstuc</code>	110	firstplural	84, 90, 135, 172, 179, 202
<code>\makeglossaries</code>	72, 77–80	long	88, 92, 180
<code>\makeglossary</code>	73	longplural	89, 92, 180
<code>makeindex</code>	202	name	82, 89, 105
<code>makeindex</code>	72	plural	83, 90, 135, 170, 171, 178
<code>\makenoidxglossaries</code>	78	see	9, 17, 19, 73
<code>\MakeTextUppercase</code>	167	short	87, 91, 116
<code>\MakeUppercase</code>	166, 167	shortplural	88, 92, 116
<code>\marg</code>	199	symbol	85, 90
<code>\markboth</code>	166	symbolplural	85, 86, 91
<code>\markright</code>	166	text 47, 82, 83, 89, 90, 135, 136, 169, 170, 177	
<code>\maxdimen</code>	24	<code>\newif</code>	104, 135
<code>\mbox</code>	200	<code>\newlength</code>	191
<code>\medskip</code>	80	<code>\newnum</code>	11
<code>\MessageBreak</code>	19, 23, 59, 68, 74, 132	<code>\newrobustcmd</code>	42, 43, 51, 52, 62,
			109, 110, 120–129, 166, 168–176, 192–198
		<code>\newsym</code>	11

<code>\newterm</code>	96	<code>all</code>	11
<code>\newtoks</code>	116	<code>false</code>	11
<code>\newwrite</code>	72	<code>none</code>	11
<code>\NoCaseChange</code>	53–55, 168–176	<code>true</code>	11
<code>\noexpand</code> ...	13, 70, 76, 77, 105, 106, 118, 184	<code>style</code>	13
<code>\nofiles</code>	79	<code>stylemods</code>	13
<code>\noindent</code>	80	<code>symbols</code>	10, 97
<code>\nopostdesc</code>	21, 22, 96	<code>undefaction</code>	15, 16
<code>\nr</code>	5, 7, 8, 11, 12	<code>error</code>	5
<code>\ns@GLSxtrfull</code>	121	<code>warn</code>	5
<code>\ns@GLsxtrfull</code>	121	<code>\PackageError</code> .	5, 13, 19, 22, 23, 42, 43, 51, 52, 57–59, 65, 67, 69, 71–74, 132–134, 184
<code>\ns@glxtrfull</code>	120	<code>\PackageWarning</code>	9
<code>\ns@GLSxtrfullpl</code>	123	<code>\PackageWarningNoLine</code>	9
<code>\ns@GLsxtrfullpl</code>	122	<code>\par</code>	79, 80, 190, 191, 200
<code>\ns@glxtrfullpl</code>	122	<code>\parindent</code>	191, 200
<code>\ns@GLSxtrlong</code>	126	<code>\PassOptionsToPackage</code>	4
<code>\ns@GLsxtrlong</code>	125	<code>\preto</code>	45
<code>\ns@glxtrlong</code>	125	<code>\printabbreviations</code>	9
<code>\ns@GLSxtrlongpl</code>	129	<code>\printglossaries</code>	73, 79
<code>\ns@GLsxtrlongpl</code>	128, 129	<code>\printglossary</code>	9, 73, 79
<code>\ns@glxtrlongpl</code>	128	<code>\printglossary options</code>	
<code>\ns@GLSxtrshort</code>	124	<code>nonumberlist</code>	27
<code>\ns@GLsxtrshort</code>	124	<code>\printnoidxglossaries</code>	79
<code>\ns@glxtrshort</code>	123	<code>\printnoidxglossary</code>	79
<code>\ns@GLSxtrshortpl</code>	127	<code>\printnumbers</code>	11, 97
<code>\ns@GLsxtrshortpl</code>	127	<code>\printsymbols</code>	11, 97
<code>\ns@glxtrshortpl</code>	126	<code>\printunsrtglossary</code>	81
<code>\null</code>	12	<code>\ProcessOptions</code>	184
<code>\number</code>	64–67	<code>\ProcessOptionsX</code>	13
<code>\numexpr</code>	64, 65, 67	<code>\protect</code> .	53–56, 89–92, 118, 119, 135–141, 143, 144, 146–157, 159, 162–164, 168–176
O			
<code>\or</code>	5, 7, 8, 11, 12, 19	<code>\protected@csedef</code>	191
<code>\org@glossaryentrynumbers</code>	24, 25	<code>\protected@csxdef</code>	191
P			
<code>\p@gl@s@hyp@opt</code>	46	<code>\protected@edef</code>	23, 70, 105, 118
package options:		<code>\protected@write</code>	6, 26, 72, 81
abbreviations	9, 10	<code>\providecommand</code>	9, 26, 43, 59, 68, 72, 76, 77, 184
accsupp	12, 82	<code>\ProvidesFile</code>	182
acronym	10	<code>\ProvidesPackage</code>	4, 184
automake	74, 78	Q	
docdef	8, 18, 19, 58, 65	<code>\quotechar</code>	107
false	19	R	
restricted	9	<code>\raggedright</code>	187, 189
true	19	<code>\relax</code>	5, 7, 8, 10–12, 14, 19, 23, 24, 26, 29, 46, 51, 59, 60, 67, 68, 73, 75, 105, 106, 108, 110, 112, 116, 117, 192–201
nonumberlist	24	<code>resize package</code>	151
numbers	10		
record	6, 7, 28, 80		
shortcuts	11		

<code>\renewcommand</code>	7–16, 19, 20, 22, 23, 25–29, 42, 44, 45, 47, 51, 56–59, 63, 65–67, 69–74, 76, 77, 96, 98–103, 108– 112, 119, 133, 135–165, 185–190, 200, 201	<code>\tabletail</code>	188, 189
<code>\renewenvironment</code>	185–189, 199	<code>\tabularnewline</code>	185–190
<code>\renewglossarystyle</code>	185–189, 199	<code>\TeX</code>	78
<code>\renewrobustcmd</code>	29	<code>\texorpdfstring</code>	53–55, 176–182
<code>\RequireGlossariesExtraLang</code>	182	<code>textcase package</code>	165
<code>\RequirePackage</code>	4, 12–14, 184	<code>\textsc</code>	147
<code>\reserved@a</code>	114	<code>\textsmaller</code>	151
<code>\reserved@b</code>	114	<code>\texttt</code>	77–80
<code>\reserved@d</code>	114	<code>\the</code>	70, 108, 118, 135–139, 141, 143, 144, 146, 147, 156–159, 162–164
<code>\RestoreAcronyms</code>	71	<code>\theindex</code>	104
<code>\romannumeral</code>	191, 192, 199	<code>\this@dialect</code>	182
		<code>\toks@</code>	108
S		U	
<code>\s@gl@s@hyp@opt</code>	46	<code>\undef</code>	109
<code>\s@gl@xtr@enabletagging</code>	109	<code>\underline</code>	110
<code>\seename</code>	17	<code>\unskip</code>	18, 185
<code>\setabbreviationstyle</code>	71, 136, 144	<code>\usepackage</code>	79, 80
<code>\setacronymstyle</code>	70, 71		
<code>\SetGenericNewAcronym</code>	71	V	
<code>\setglossarystyle</code>	13, 185, 201	<code>\val</code>	5, 7, 8, 11, 12
<code>\setkeys</code>	6, 13, 14, 30, 45, 69, 116, 117		
<code>\setlength</code>	24, 191, 200	W	
<code>\settowidth</code>	72, 191–199	<code>\warn@nomakeglossaries</code>	73
<code>\setupglossaries</code>	4, 14	<code>\warn@noprintglossary</code>	73
<code>\sfcode</code>	112, 190	<code>\write</code>	59, 68, 72, 76, 77
<code>\space</code>	5, 20, 23, 57–59, 65, 67–69, 71–73, 77, 80, 112, 119, 136, 190, 191, 199		
<code>\spacefactor</code>	112, 117, 190	X	
<code>\string</code>	5, 6, 19, 20, 23, 26, 42, 43, 51, 52, 57–59, 65, 67–69, 71–74, 76–81, 100, 102, 103, 105	<code>\xcapitalisewords</code>	98
<code>\strut</code>	185–190	<code>\xifinlist</code>	64
<code>\subglossentry</code>	185–190, 200	<code>xindy</code>	202
		<code>xindy</code>	72
T		<code>xkeyval package</code>	4
<code>\tablehead</code>	188, 189	<code>\XKV@checkchoice</code>	27
		<code>\XKV@plfalse</code>	27
		<code>\XKV@resa</code>	27
		<code>\XKV@sttrue</code>	27