

`glossaries-extra.sty v1.07: documented code`

Nicola L.C. Talbot

Dickimaw Books

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

2016-08-15

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	12
1.3 Modifications to Commands Provided by glossaries	12
1.3.1 Existence Checks	12
1.3.2 Document Definitions	16
1.3.3 Existing Glossary Style Modifications	20
1.3.4 Entry Formatting, Hyperlinks and Indexing	24
1.3.5 Entry Counting	51
1.3.6 Acronym Modifications	64
1.3.7 Indexing and Displaying Glossaries	67
1.4 Integration with glossaries-accsupp	76
1.5 Categories	86
1.6 Abbreviations	109
1.6.1 Abbreviation Styles Setup	126
1.6.2 Predefined Styles (Default Font)	129
1.6.3 Predefined Styles (Small Capitals)	142
1.6.4 Predefined Styles (Fake Small Capitals)	146
1.6.5 Predefined Styles (Emphasized)	149
1.6.6 Predefined Styles (User Parentheses Hook)	155
1.7 Using Entries in Headings	159
1.8 Multi-Lingual Support	176
2 Style Adjustments (glossaries-extra-stylemods.sty)	178
2.1 Package Initialisation	178
2.2 List-Like Styles	179
2.3 Longtable Styles	179
2.4 Long Ragged Styles	180
2.5 Supertabular Styles	182
2.6 Super Ragged Styles	183
2.7 Inline Style	184
2.8 Tree Styles	184
Glossary	196
Change History	197
Index	205

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2016/08/15 v1.07 (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.

```
35 \define@choicekey{glossaries-extra.sty}{undefaction}[\val\nr]%
36 {warn,error}%
37 {%
38   \ifcase\nr\relax
39     \renewcommand*{\glxtrundefaction}[2]{%
40       \@glxtrundeftag\GlossariesExtraWarning{##1}%
41     }%
42     \renewcommand*{\glxtr@warnonexistsordo}[1]{%
43       \GlossariesExtraWarning{glossaries-extra}{%
44         \string##1\space hasn't been defined, so
45         some errors won't be converted to warnings.
46         (This most likely means your version of
47         glossaries.sty is below version 4.19.))}%
48     }%
49   \or
50     \renewcommand*{\glxtrundefaction}[2]{%
51       \@glxtrundeftag\PackageError{glossaries-extra}{##1}{##2}%
52     }%
53     \renewcommand*{\glxtr@warnonexistsordo}[1]{}%
54   \fi
55 }
```

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

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

```
56 \newcount\@glxtr@docdefval
```

Need to provide conditional commands that are backward compatible:

`\if@glxtrdocdef`

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

lsxtrdocdeftrue

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

lsxtrdocdeffalse

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

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

```
60 \define@choicekey{glossaries-extra.sty}{docdef}[\val\nr]%  
61 {false,true,restricted}[true]%  
62 {%  
63   \@glxtr@docdefval=\nr\relax  
64   \ifnum\@glxtr@docdefval=2\relax  
65     \renewcommand*{\@glsdofexistsorwarn}{\glsdofexists}%  
66   \fi  
67 }
```

ocdefrestricted

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

oifexistsorwarn

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

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

indexcrossrefs

Automatically index cross references at the end of the document

```
70 \define@boolkey{glossaries-extra.sty}[\@glxtr]{indexcrossrefs}[true]{%  
71   \if@glxtrindexcrossrefs  
72   \else  
73     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%  
74   \fi  
75 }
```

Switch off since this can increase the build time.

```
76 \@glxtrindexcrossrefsfalse
```

But allow see key to switch it on automatically.

oindexcrossrefs

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

iesExtraWarning

Allow users to suppress warnings.

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

raWarningNoLine

Allow users to suppress warnings.

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

```

81 \@glxstr@declareoption{nowarn}{%
82   \let\GlossariesExtraWarning\@gobble
83   \let\GlossariesExtraWarningNoLine\@gobble
84   \glxstr@doption{nowarn}%
85 }

glxstrabbrvtype Glossary type for abbreviations.
86 \newcommand*{\glxstrabbrvtype}{\glstypedefaulttype}

abbreviationsdef Set by abbreviations option.
87 \newcommand*{\@glxstr@abbreviationsdef}{}

abbreviationsdef
88 \newcommand*{\@glxstr@doabbreviationsdef}{%
89   \ifpackageloaded{babel}%
90     {\providecommand{\abbreviationsname}{\acronymname}}%
91     {\providecommand{\abbreviationsname}{Abbreviations}}%
92     \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
93     \renewcommand*{\glxstrabbrvtype}{abbreviations}%
94     \newcommand*{\printabbreviations}[1][1]{%
95       \printglossary[type=\glxstrabbrvtype,##1]%
96     }%
97     \disable@keys{glossaries-extra.sty}{abbreviations}%

    If the acronym option hasn't been used, change \acronymtype to \glxstrabbrvtype.
98   \ifglsacronym
99   \else
100     \renewcommand*{\acronymtype}{\glxstrabbrvtype}%
101   \fi
102 }%

abbreviations If abbreviations, create a new glossary type for abbreviations.
103 \@glxstr@declareoption{abbreviations}{%
104   \let\@glxstr@abbreviationsdef\@glxstr@doabbreviationsdef
105 }

iationShortcuts Enable shortcut commands for the abbreviations. Unlike the analogous command provided
by glossaries, this uses \newcommand instead of \let as a safety feature.
106 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
107   \newcommand*{\ab}{\cglsl}%
108   \newcommand*{\abp}{\cglspl}%
109   \newcommand*{\as}{\glxstrshort}%
110   \newcommand*{\asp}{\glxstrshortpl}%
111   \newcommand*{\al}{\glxstrlong}%
112   \newcommand*{\alp}{\glxstrlongpl}%
113   \newcommand*{\af}{\glxstrfull}%
114   \newcommand*{\afp}{\glxstrfullpl}%
115   \newcommand*{\Ab}{\cGls}%
116   \newcommand*{\Abp}{\cGlspl}%

```

```

117 \newcommand*{\As}{\Glsxtrshort}%
118 \newcommand*{\Asp}{\Glsxtrshortpl}%
119 \newcommand*{\Al}{\Glsxtrlong}%
120 \newcommand*{\Alp}{\Glsxtrlongpl}%
121 \newcommand*{\Af}{\Glsxtrfull}%
122 \newcommand*{\Afp}{\Glsxtrfullpl}%
123 \newcommand*{\AB}{\cGLS}%
124 \newcommand*{\ABP}{\cGLSp1}%
125 \newcommand*{\AS}{\GLSxtrshort}%
126 \newcommand*{\ASP}{\GLSxtrshortpl}%
127 \newcommand*{\AL}{\GLSxtrlong}%
128 \newcommand*{\ALP}{\GLSxtrlongpl}%
129 \newcommand*{\AF}{\GLSxtrfull}%
130 \newcommand*{\AFP}{\GLSxtrfullpl}%
131 \newcommand*{\newabbr}{\newabbreviation}%

Disable this command after it's been used.
132 \let\GlsXtrDefineAbbreviationShortcuts\relax
133 }

```

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

```

134 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
135 \newcommand*{\newentry}{\newglossaryentry}%
136 \ifdef\printsymbols
137 {%
138 \newcommand*{\newsym}{\glxtrnewsymbol}%
139 }{}%
140 \ifdef\printnumbers
141 {%
142 \newcommand*{\newnum}{\glxtrnewnumber}%
143 }{}%
144 \let\GlsXtrDefineOtherShortcuts\relax
145 }

```

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.

```

146 \newcommand*{\@glxtr@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.

```

147 \define@choicekey{glossaries-extra.sty}{shortcuts}[\val\nr]%
148 {acronyms,acro,abbreviations,abbr,other,all,true,none,false}[true]{%
149 \ifcase\nr\relax % acronyms
150 \renewcommand*{\@glxtr@setupshortcuts}{%

```



```

151     \glsacrshortcutstrue
152     \DefineAcronymSynonyms
153 }%
154 \or % acro
155     \renewcommand*{\@glsxtr@setupshortcuts}{%
156         \glsacrshortcutstrue
157         \DefineAcronymSynonyms
158     }%
159 \or % abbreviations
160     \renewcommand*{\@glsxtr@setupshortcuts}{%
161         \GlsXtrDefineAbbreviationShortcuts
162     }%
163 \or % abbr
164     \renewcommand*{\@glsxtr@setupshortcuts}{%
165         \GlsXtrDefineAbbreviationShortcuts
166     }%
167 \or % other
168     \renewcommand*{\@glsxtr@setupshortcuts}{%
169         \GlsXtrDefineOtherShortcuts
170     }%
171 \or % all
172     \renewcommand*{\@glsxtr@setupshortcuts}{%
173         \glsacrshortcutstrue
174         \DefineAcronymSynonyms
175         \GlsXtrDefineAbbreviationShortcuts
176         \GlsXtrDefineOtherShortcuts
177     }%
178 \or % true
179     \renewcommand*{\@glsxtr@setupshortcuts}{%
180         \glsacrshortcutstrue
181         \DefineAcronymSynonyms
182         \GlsXtrDefineAbbreviationShortcuts
183         \GlsXtrDefineOtherShortcuts
184     }%
185 \else % none, false
186     \renewcommand*{\@glsxtr@setupshortcuts}{}%
187 \fi
188 }

```

lsxtr@doaccsupp

```

189 \newcommand*{\@glsxtr@doaccsupp}{}

```

accsupp If accsupp, load glossaries-accsupp package.

```

190 \@glsxtr@declareoption{accsupp}{%
191     \renewcommand*{\@glsxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}

```

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

```

192 \newcommand{\glsxtrNoGlossaryWarning}[1]{%

```

```

193 \@glsxtr@defaultnoglossarywarning{#1}%
194 }

```

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

```

195 \define@choicekey{glossaries-extra.sty}{nomissingglstext}[\val\nr]{%
196 {true,false}[true]{%
197 \ifcase\nr\relax % true
198 \renewcommand{\glsxtrNoGlossaryWarning}[1]{%
199 \null
200 }%
201 \else % false
202 \renewcommand{\glsxtrNoGlossaryWarning}[1]{%
203 \@glsxtr@defaultnoglossarywarning{#1}%
204 }%
205 \fi
206 }

```

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

xtr@redefstyles

```

207 \newcommand*{\@glsxtr@redefstyles}{}

```

stylemods

```

208 \define@key{glossaries-extra.sty}{stylemods}{%
209 \ifblank{#1}%
210 {%
211 \renewcommand*{\@glsxtr@redefstyles}{%
212 \RequirePackage{glossaries-extra-stylemods}}%
213 }%
214 {%
215 \renewcommand*{\@glsxtr@redefstyles}{}%
216 \@for\@glsxtr@tmp:=#1\do{%
217 \IfFileExists{glossary-\@glsxtr@tmp.sty}%
218 {%
219 \eappto\@glsxtr@redefstyles{%
220 \noexpand\RequirePackage{glossary-\@glsxtr@tmp}}%
221 }%
222 {%
223 \PackageError{glossaries-extra}%
224 {Glossaries style package ‘glossary-\@glsxtr@tmp.sty’
225 doesn’t exist (did you mean to use the ‘style’ key?)}%
226 {The list of values (#1) in the ‘stylemods’ key should
227 match the glossary-xxx.sty files provided with
228 glossaries.sty}%
229 }%
230 }%
231 \appto\@glsxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
232 }%
233 }

```

glsxtr@do@style

```
234 \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.

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

```
236 \renewcommand*{\@glsxtr@do@style}{%
```

Set this as the default style:

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

Set this style:

```
238 \setglossarystyle{#1}%
```

```
239 }%
```

```
240 }
```

Pass all other options to glossaries.

```
241 \DeclareOptionX*{%
```

```
242 \expandafter\glsxtr@doooption\expandafter{\CurrentOption}}
```

Process options.

```
243 \ProcessOptionsX
```

Load glossaries if not already loaded.

```
244 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
245 \@glsxtr@doaccsupp
```

Define abbreviations glossaries if required.

```
246 \@glsxtr@abbreviationsdef
```

```
247 \let\@glsxtr@abbreviationsdef\relax
```

Setup shortcuts if required.

```
248 \@glsxtr@setupshortcuts
```

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

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

Now define the user command:

```
250 \newcommand*{\glossariesextrasetup}[1]{%
```

```
251 \let\@glsxtr@setupshortcuts\relax
```

```
252 \setkeys{glossaries-extra.sty}{#1}%
```

```
253 \@glsxtr@abbreviationsdef
```

```
254 \let\@glsxtr@abbreviationsdef\relax
```

```
255 \@glsxtr@setupshortcuts
```

```
256 }
```

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

```
257 \AtBeginDocument{%
258   \disable@keys{glossaries-extra.sty}{abbreviations,docdef}%
259   \def\@glsxtrundeftag{\glsxtrundeftag}%
260 }
```

1.2 Extra Utilities

`\glsxtrifemptyglossary`

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

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

```
261 \newcommand{\glsxtrifemptyglossary}[3]{%
262   \ifglossaryexists{#1}%
263   {%
264     \ifcsstring{glolist@#1}{,}{#2}{#3}%
265   }%
266   {%
267     \glsxtrundefaction{Glossary type '#1' doesn't exist}{}%
268     #2%
269   }%
270 }
```

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

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

```
271 \renewcommand{\glsdoifexists}[2]{%
272   \ifglsentryexists{#1}{#2}%
273   {%
274     \glsxtrundefaction{Glossary entry '\glsdetoklabel{#1}'
275       has not been defined}{You need to define a glossary entry before
276       you can reference it.}%
277   }%
278 }
```

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

```
279 \renewcommand{\glsdoifnoexists}[2]{%
280   \ifglstryexists{#1}{%
281     \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
282       has already been defined}{}}{#2}%
283 }
```

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

```
284 \ifdefined\glsdoifexistsordo
285 {%
286   \renewcommand{\glsdoifexistsordo}[3]{%
287     \ifglstryexists{#1}{#2}%
288     {%
289       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
290         has not been defined}{You need to define a glossary entry
291         before you can use it.}%
292       #3%
293     }%
294   }%
295 }
296 {%
297   \glstr@warnonexistsordo\glsdoifexistsordo
298   \newcommand{\glsdoifexistsordo}[3]{%
299     \ifglstryexists{#1}{#2}%
300     {%
301       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
302         has not been defined}{You need to define a glossary entry
303         before you can use it.}%
304       #3%
305     }%
306   }%
307 }
```

`glsarynoexistsordo` Similarly for `\doifglossarynoexistsordo`.

```
308 \ifdefined\doifglossarynoexistsordo
309 {%
310   \renewcommand{\doifglossarynoexistsordo}[3]{%
311     \ifglossaryexists{#1}%
312     {%
313       \glstrundefaction{Glossary type '#1' already exists}{}%
314       #3%
315     }%
316     {#2}%
317   }%
318 }
319 {%
320   \glstr@warnonexistsordo\doifglossarynoexistsordo
321   \newcommand{\doifglossarynoexistsordo}[3]{%

```

```

322 \ifglossaryexists{#1}%
323 {%
324 \glstrundefaction{Glossary type ‘#1’ already exists}{}%
325 #3%
326 }%
327 {#2}%
328 }%
329 }
330

```

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

```

331 \appto\@newglossaryentryposthook{%
332 \ifdefvoid\@glo@see
333 {\csxdef{glo@\@glo@label @see}{}}%
334 {%
335 \csxdef{glo@\@glo@label @see}{\@glo@see}%
336 \@glstr@autoindexcrossrefs
337 }%
338 }
339 \appto\@gls@keymap{,{see}{see}}

```

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

```

340 \newcommand*{\glstrusesee}[1]{%
341 \glsdoifexists{#1}%
342 {%
343 \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
344 \ifdefempty\@glo@see
345 {}%
346 {%
347 \expandafter\glstr@usesee\@glo@see\@end@glstr@usesee
348 }%
349 }%
350 }

```

`\glstr@usesee`

```

351 \newcommand*{\glstr@usesee}[1][\seename]{%
352 \@glstr@usesee[#1]%
353 }

```

`\@glstr@usesee`

```

354 \def\@glstr@usesee[#1]#2\@end@glstr@usesee{%
355 \glstruseseeformat{#1}{#2}%
356 }

```

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

```

357 \newcommand*{\glstruseseeformat}[2]{%
358 \glsseeformat[#1]{#2}{}%
359 }

```

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

```
360 \AtEndDocument{\ifglxtrindexcrossrefs\glxtraddallcrossrefs\fi}
```

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

```
361 \newcommand*{\glxtraddallcrossrefs}{%
362   \forallglossaries{\@glo@type}%
363   {%
364     \forlgsentries[\@glo@type]{\@glo@label}%
365     {%
366       \ifglxsused{\@glo@label}{\@glxtr@addunusedxrefs{\@glo@label}}{%
367       }%
368     }%
369 }
```

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

```
370 \newcommand*{\@glxtr@addunusedxrefs}[1]{%
371   \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@see}%
372   \ifdefvoid\@glo@see
373   {%
374   {%
375     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
376   }%
377 }
```

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

```
378 \newcommand*{\glxtr@addunused}[1][]{%
379   \@glxtr@addunused
380 }
```

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

```
381 \def\@glxtr@addunused#1\@end@glxtr@addunused{%
382   \@for\@glxtr@label:=#1\do
383   {%
384     \ifglxsused{\@glxtr@label}{}%
385     {%
386       \glxtradd[format=glxtrunusedformat]{\@glxtr@label}%
387       \glxtrunset{\@glxtr@label}%
388       \@glxtr@addunusedxrefs{\@glxtr@label}%
389     }%
390   }%
391 }
```

xtrunusedformat

```
392 \newcommand*{\glxtrunusedformat}[1]{\unskip}
```

1.3.2 Document Definitions

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

```
393 \let\glstr@orgmakenoidxglossaries\makenoidxglossaries
394 \renewcommand{\makenoidxglossaries}{%
395   \glstr@orgmakenoidxglossaries
396   \if@glstrdocdefrestricted
```

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

```
397   \renewcommand*{\@gls@reference}[3]{%
398     \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
399     \ifinlistcs{##2}{\@glsref@##1}%
400     {}%
401     {\listcsgadd{\glsref@##1}{##2}}%
402     \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
403     {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
404     {}%
405     \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
406   }%
407 \else
```

Disable document definitions.

```
408   \@glstrdocdeffalse
409 \fi
410 \disable@keys{glossaries-extra.sty}{docdef}%
411 }
```

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

```
412 \renewcommand*{\gls@defdocnewglossaryentry}{%
413   \ifcase\@glstr@docdefval
414     docdef=false:
415     \renewcommand*{\newglossaryentry}[2]{%
416       \PackageError{glossaries-extra}{Glossary entries must
417       be \MessageBreak defined in the preamble with \MessageBreak
418       package option 'docdef=false'\MessageBreak(consider using
419       'docdef=restricted')}{Move your glossary definitions to
420       the preamble. You can also put them in a \MessageBreak separate file
421       and load them with \string\loadglsentries.}%
422     }%
423   \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.

```
423   \let\gls@checkseeallowed\relax
424   \let\newglossaryentry\new@glossaryentry
425 \or
```


Restricted mode just needs to allow the see value.

```
426 \let\gls@checkseeallowed\relax
427 \fi
428 }%
```

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`

```
429 \newcommand*{\GlsXtrEnableOnTheFly}{%
430 \ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
431 }
```

`\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.)

```
432 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
433 \renewcommand*{\glsdetoklabel}[1]{%
434 \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
435 {%
436 \expandafter\detokenize\expandafter{##1}%
437 }%
438 {\detokenize{##1}}}%
439 }%
440 \@GlsXtrEnableOnTheFly
441 }
442 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
443 \expandafter\if\glsbackslash#1%
444 #3%
445 \else
446 #4%
447 \fi
448 }
```

`\sXtrstarflywarn`

```
449 \newcommand*{\glsxtrstarflywarn}{%
450 \GlossariesExtraWarning{Experimental starred version of
451 \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
452 read the warnings in the glossaries-extra user manual)}}%
453 }
```

`\rEnableOnTheFly`

```
454 \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:

```

\glstrcat
455 \newcommand*\glstrcat{general}

\glstr
456 \newcommand*\glstr[1] [] {%
457 \def\glstr@keylist{##1}%
458 \@glstr
459 }

\@glstr
460 \newcommand*\@glstr[2] [] {%
461 \ifglstryexists{##2}%
462 {%
463 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
464 }%
465 {%
466 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
467 description={\nopostdesc},##1}%
468 }%
469 \expandafter\gls\expandafter[\glstr@keylist]{##2}%
470 }

\Glsxtr
471 \newcommand*\Glsxtr[1] [] {%
472 \def\glstr@keylist{##1}%
473 \@Glsxtr
474 }

\@Glsxtr
475 \newcommand*\@Glsxtr[2] [] {%
476 \ifglstryexists{##2}%
477 {%
478 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
479 }%
480 {%
481 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
482 description={\nopostdesc},##1}%
483 }%
484 \expandafter\Gls\expandafter[\glstr@keylist]{##2}%
485 }

\glstrpl
486 \newcommand*\glstrpl[1] [] {%
487 \def\glstr@keylist{##1}%
488 \@glstrpl
489 }

\@glstrpl

```

```

490 \newcommand*{\@glxstrpl}[2][ ]{%
491   \ifglentryexists{##2}%
492   {%
493     \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
494   }%
495   {%
496     \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
497       description={\nopostdesc},##1}%
498   }%
499   \expandafter\glsp\expandafter[\glxtr@keylist]{##2}%
500 }

```

\Glsxtrpl

```

501 \newcommand*{\Glsxtrpl}[1][ ]{%
502   \def\glxtr@keylist{##1}%
503   \@Glsxtrpl
504 }

```

\@Glsxtrpl

```

505 \newcommand*{\@Glsxtrpl}[2][ ]{%
506   \ifglentryexists{##2}
507   {%
508     \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
509   }%
510   {%
511     \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
512       description={\nopostdesc},##1}%
513   }%
514   \expandafter\glsp\expandafter[\glxtr@keylist]{##2}%
515 }

```

\GlsXtrWarning

```

516 \newcommand*{\GlsXtrWarning}[2]{%
517   \def\glxtr@optlist{##1}%
518   \@onelevel@sanitize\glxtr@optlist
519   \GlossariesExtraWarning{The options ‘\glxtr@optlist’ have
520     been ignored for entry ‘##2’ as it has already been defined}%
521 }

```

Disable commands after the glossary:

```

522 \let\@glxtr@orgprintglossary\@printglossary
523 \renewcommand\@printglossary[2]{%
524   \@glxtr@orgprintglossary{##1}{##2}%
525   \def\@glxtr{\@glxtr@disabledflycommand\glxtr}%
526   \def\@glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
527   \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
528   \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
529 }

```

abledflycommand

```

530 \newcommand*{\@glxtr@disabledflycommand}[1]{%
531   \PackageError{glossaries-extra}%
532   {\string##1\space can't be used after any of the \MessageBreak
533    glossaries have been displayed}%
534   {The on-the-fly commands enabled by
535    \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
536    before the glossaries. If you want to use any entries \MessageBreak
537    after any of the glossaries, you must use the standard \MessageBreak
538    method of first defining the entry and then using the \MessageBreak
539    entry with commands like \string\gls}%
540   \@glxtr@disabledflycommand
541 }%
542 \newcommand*{\@glxtr@disabledflycommand}[2][\{##2}

```

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

```

543 \let\GlsXtrEnableOnTheFly\relax
544 }
545 \@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.

```

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

```

Modify \setglossarystyle to set the above.

etglossarystyle

```

547 \renewcommand*{\setglossarystyle}[1]{%
548   \ifcsundef{@glsstyle@#1}%
549   {%
550     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
551   }%
552   {%
553     \csname @glsstyle@#1\endcsname
554     \protected@edef\@glxtr@current@style{#1}%
555   }%
556   \ifx\@glossary@default@style\relax
557     \protected@edef\@glossary@default@style{#1}%
558   \fi
559 }

```

In case we have an old version of glossaries:

```

560 \ifdef\@glossary@default@style
561 {}

```

```

562 {%
563   \let\@glossary@default@style\relax
564 }

```

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

```

565 \ifdef\glslistdottedwidth
566 {%
567   \ifdim\glslistdottedwidth=.5\hsize
568     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
569     \AtBeginDocument{%
570       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
571         \setlength{\glslistdottedwidth}{.5\columnwidth}%
572       \fi
573     }%
574   \fi
575 }
576 {}%

```

Similarly for \glsdescwidth:

\glsdescwidth

```

577 \ifdef\glsdescwidth
578 {%
579   \ifdim\glsdescwidth=.6\hsize
580     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
581     \AtBeginDocument{%
582       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
583         \setlength{\glsdescwidth}{.6\columnwidth}%
584       \fi
585     }%
586   \fi
587 }
588 {}%

```

and for \glspagelistwidth:

lspagelistwidth

```

589 \ifdef\glspagelistwidth
590 {%
591   \ifdim\glspagelistwidth=.1\hsize
592     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
593     \AtBeginDocument{%
594       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
595         \setlength{\glspagelistwidth}{.1\columnwidth}%
596       \fi
597     }%
598   \fi
599 }
600 {}%

```

aryentrynumbers Has the nonumberlist option been used?

```
601 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
602 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
603   \glsnonumberlistfalse
604   \renewcommand*{\glossaryentrynumbers}[1]{%
605     \ifglsentryexists{\glscurrententrylabel}%
606     {%
607       \@glsxtrpreloctag
608       \GlsXtrFormatLocationList{#1}%
609       \@glsxtrpostloctag
610       \gls@save@numberlist{#1}%
611     }{}%
612   }%
613 \else
614   \glsnonumberlisttrue
615   \renewcommand*{\glossaryentrynumbers}[1]{%
616     \ifglsentryexists{\glscurrententrylabel}%
617     {%
618       \gls@save@numberlist{#1}%
619     }{}%
620   }%
621 \fi
```

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

```
622 \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

```
623 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
624   \let\@glsxtrpreloctag\@glsxtrpreloctag
625   \let\@glsxtrpostloctag\@glsxtrpostloctag
626   \renewcommand*{\@glsxtr@pagetag}{#1}%
627   \renewcommand*{\@glsxtr@pagetag}{#2}%
628   \renewcommand*{\@glsxtr@savepreloctag}[2]{%
629     \csgdef{\@glsxtr@preloctag@##1}{##2}%
630   }%
631   \renewcommand*{\@glsxtr@doloctag}{%
632     \ifcsundef{\@glsxtr@preloctag\@glscurrententrylabel}%
633     {%
634       \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
635         Rerun required}%
636     }%
637     {%
```

```

638      \csuse{@glxstr@preloctag@\glscurrententrylabel}%
639    }%
640  }%
641 }
642 \@onlypreamble\GlsXtrEnablePreLocationTag

glxstrpreloctag
643 \newcommand*{\@@glxstrpreloctag}{%
644   \let\@glxstr@org@delimN\delimN
645   \let\@glxstr@org@delimR\delimR
646   \let\@glxstr@org@glsgignore\glsgignore
647   \gdef is required as the delimiters may occur inside a scope.
648   \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
649   \renewcommand*{\delimN}{%
650     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
651     \@glxstr@org@delimN}%
652   \renewcommand*{\delimR}{%
653     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
654     \@glxstr@org@delimR}%
655   \renewcommand*{\glsgignore}[1]{%
656     \gdef\@glxstr@thisloctag{\relax}%
657     \@glxstr@org@glsgignore{##1}}%
658   \@glxstr@doloctag
659 }

glxstrpreloctag
660 \newcommand*{\@glxstrpreloctag}{%

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

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

lsxtrpostloctag
663 \newcommand*{\@@glxtrpostloctag}{%
664   \let\delimN\@glxstr@org@delimN
665   \let\delimR\@glxstr@org@delimR
666   \let\glsgignore\@glxstr@org@glsgignore
667   \protected@write\@auxout{%
668     {\string\@glxstr@savepreloctag{\glscurrententrylabel}\@glxstr@thisloctag}}%
669 }

lsxtrpostloctag
670 \newcommand*{\@glxtrpostloctag}{%

```

lsxtr@preloctag

```
670 \newcommand*{\@glxtr@savepreloctag}[2]{%
671 \protected@write\@auxout{}{%
672   \string\providecommand\string\@glxtr@savepreloctag[2]{}}
```

glxtr@doloctag

```
673 \newcommand*{\@glxtr@doloctag}{}
```

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

```
674 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
675   \XKV@plfalse
676   \XKV@sttrue
677   \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
678   {%
679     \csname glsnonumberlist\XKV@resa\endcsname
680     \ifglsnonumberlist
681       \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
682     \else
683       \def\glossaryentrynumbers##1{%
684         \@glxtr@preloctag
685         \GlsXtrFormatLocationList{##1}%
686         \@glxtr@postloctag
687         \gls@save@numberlist{##1}}%
688     \fi
689   }%
690 }
```

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.

```
691 \renewcommand*{\glsentryfmt}{%
692   \ifglshasshort{\glslabel}{\glsssetabbrvfmt{\glscategory{\glslabel}}}%
693   \glsifregular{\glslabel}%
694   {\glxtrregularfont{\glsgenentryfmt}}%
695   {%
696     \ifglshasshort{\glslabel}%
697     {\glxtrgenabbrvfmt}%
698     {\glxtrregularfont{\glsgenentryfmt}}%
699   }%
700 }
```


sxtrregularfont Font used for regular entries.

```
701 \newcommand*{\glxsxtrregularfont}[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 `\glstryfmt`.

@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.

```
702 \renewcommand{\@gls@field@link}[4][]{%}
```

```
703   \glsdoifexists{#3}%
```

```
704   {%
```

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).

```
705     \let\glxsxtrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
```

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

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

```
708     \@glxsxtr@field@linkdefs
```

```
709     #1%
```

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

```
711     \let\ifKV@glslink@hyper\glxsxtrorg@ifKV@glslink@hyper
```

```
712   }%
```

```
713   \glspostlinkhook
```

```
714 }
```

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

```
715 \newcommand*{\@glxsxtr@field@linkdefs}{%
```

```
716   \let\glsxtrifwasfirstuse\@secondoftwo
```

```
717   \let\glsifplural\@secondoftwo
```

```
718   \let\glscapscase\@firstofthree
```

```
719   \let\glsinsert\@empty
```

```
720 }
```

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

assignfieldfont

```
721 \newcommand*{\glxsxtrassignfieldfont}[1]{%
```

```
722   \ifglshasshort{#1}%
```

```
723   {%
```

```
724     \glsssetabbrvfmt{\glscategory{#1}}%
```

```
725     \glsifregular{#1}%
```

```
726     {\let\@gls@field@font\glxsxtrregularfont}%
```

```
727     {\let\@gls@field@font\@firstofone}%
```

```
728   }%
```

```

729 {%
730   \glsifnotregular{#1}%
731   {\let\@gls@field@font\@firstofone}%
732   {\let\@gls@field@font\glsxtrregularfont}%
733 }%
734 }

```

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

```

735 \def\@glstext@#1#2[#3]{%
736   \glsxtrassignfieldfont{#2}%
737   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
738 }

```

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

```

739 \def\@GLStext@#1#2[#3]{%
740   \glsxtrassignfieldfont{#2}%
741   \@gls@field@link[\let\gls@caps@case\@thirdofthree]{#1}{#2}%
742   {\@gls@field@font{\GLS@accesstext{#2}\mfirstucMakeUppercase{#3}}}%
743 }

```

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

```

744 \def\@Glstext@#1#2[#3]{%
745   \glsxtrassignfieldfont{#2}%
746   \@gls@field@link[\let\gls@caps@case\@secondofthree]{#1}{#2}%
747   {\@gls@field@font{\GLS@accesstext{#2}#3}}%
748 }

```

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.

`\glsnohyperfirst`

```

749 \newcommand*\glsxtrchecknohyperfirst[1]{%
750   \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
751 }

```

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

```

752 \def\@glsfirst@#1#2[#3]{%
753   \glsxtrassignfieldfont{#2}%
754   \@gls@field@link
755   [\let\glsxtrifwasfirstuse\@firstoftwo
756   \glsxtrchecknohyperfirst{#2}%
757   ]{#1}{#2}%
758   {\@gls@field@font{\glsaccessfirst{#2}#3}}%
759 }

```

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

```

760 \def\@Glsfirst@#1#2[#3]{%
761   \glsxtrassignfieldfont{#2}%

```

Ensure that \Glsfirst honours the nohyperfirst attribute.

```
762 \@gls@field@link
763 [\let\glstrifwasfirstuse\@firstoftwo
764 \let\glscapscase\@secondofthree
765 \glstrchecknohyperfirst{#2}%
766 ]%
767 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
768 }
```

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

```
769 \def\@GLSfirst@#1#2[#3]{%
770 \glstrassignfieldfont{#2}%
771 \@gls@field@link
772 [\let\glstrifwasfirstuse\@firstoftwo
773 \let\glscapscase\@thirdofthree
774 \glstrchecknohyperfirst{#2}%
775 ]%
776 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
777 }
```

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

```
778 \def\@glsplural@#1#2[#3]{%
779 \glstrassignfieldfont{#2}%
780 \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
781 {\@gls@field@font{\glsaccessplural{#2}#3}}%
782 }
```

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

```
783 \def\@Glsplural@#1#2[#3]{%
784 \glstrassignfieldfont{#2}%
785 \@gls@field@link
786 [\let\glsifplural\@firstoftwo
787 \let\glscapscase\@secondofthree
788 ]%
789 {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
790 }
```

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

```
791 \def\@GLSplural@#1#2[#3]{%
792 \glstrassignfieldfont{#2}%
793 \@gls@field@link
794 [\let\glsifplural\@firstoftwo
795 \let\glscapscase\@thirdofthree
796 ]%
797 {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
798 }
```

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

```
799 \def\@glsfirstplural@#1#2[#3]{%  
800   \glstrassignfieldfont{#2}%  
  
   Ensure that \glsfirstplural honours the nohyperfirst attribute.  
801   \@gls@field@link  
802   [\let\glstrifwasfirstuse\@firstoftwo  
803   \let\glsifplural\@firstoftwo  
804   \glstrchecknohyperfirst{#2}%  
805   ]%  
806   {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%  
807 }
```

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

```
808 \def\@Glsfirstplural@#1#2[#3]{%  
809   \glstrassignfieldfont{#2}%  
  
   Ensure that \glsfirstplural honours the nohyperfirst attribute.  
810   \@gls@field@link  
811   [\let\glstrifwasfirstuse\@firstoftwo  
812   \let\glsifplural\@firstoftwo  
813   \let\glscapscase\@secondofthree  
814   \glstrchecknohyperfirst{#2}%  
815   ]%  
816   {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%  
817 }
```

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

```
818 \def\@GLSfirstplural@#1#2[#3]{%  
819   \glstrassignfieldfont{#2}%  
  
   Ensure that \glsfirstplural honours the nohyperfirst attribute.  
820   \@gls@field@link  
821   [\let\glstrifwasfirstuse\@firstoftwo  
822   \let\glsifplural\@firstoftwo  
823   \let\glscapscase\@thirdofthree  
824   \glstrchecknohyperfirst{#2}%  
825   ]%  
826   {#1}{#2}%  
827   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%  
828 }
```

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

```
829 \def\@glsname@#1#2[#3]{%  
830   \glstrassignfieldfont{#2}%  
831   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%  
832 }
```

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

```
833 \def\@Glsname@#1#2[#3]{%
```

```

834 \glstrassignfieldfont{#2}%
835 \@gls@field@link
836 [\let\glscapscase\@secondoftwo]{#1}{#2}%
837 {\@gls@field@font{\Glsaccessname{#2}#3}}%
838 }

```

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

```

839 \def\@GLSname@#1#2[#3]{%
840 \glstrassignfieldfont{#2}%
841 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
842 {#1}{#2}%
843 {\@gls@field@font{\Glsaccessname{#2}\mfirstucMakeUppercase{#3}}}%
844 }

```

\@glsdesc@

```

845 \def\@glsdesc@#1#2[#3]{%
846 \glstrassignfieldfont{#2}%
847 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
848 }

```

\@Glsdesc@ First letter uppercase version.

```

849 \def\@Glsdesc@#1#2[#3]{%
850 \glstrassignfieldfont{#2}%
851 \@gls@field@link
852 [\let\glscapscase\@secondoftwo]{#1}{#2}%
853 {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
854 }

```

\@GLSdesc@ All uppercase version.

```

855 \def\@GLSdesc@#1#2[#3]{%
856 \glstrassignfieldfont{#2}%
857 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
858 {#1}{#2}{\@gls@field@font{\Glsaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
859 }

```

@glsdescplural@ No case-changing version.

```

860 \def\@glsdescplural@#1#2[#3]{%
861 \glstrassignfieldfont{#2}%
862 \@gls@field@link
863 [\let\glscapscase\@secondoftwo
864 \let\glsifplural\@firstoftwo
865 ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
866 }

```

@Glsdescplural@ First letter uppercase version.

```

867 \def\@Glsdescplural@#1#2[#3]{%
868 \glstrassignfieldfont{#2}%
869 \@gls@field@link

```

```

870 [\let\glscapscase\@secondoftwo
871 \let\glsifplural\@firstoftwo
872 ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
873 }

```

@GLSdescplural@ All uppercase version.

```

874 \def\@GLSdesc@#1#2[#3]{%
875 \glstrassignfieldfont{#2}%
876 \@gls@field@link
877 [\let\glscapscase\@thirdoftwo
878 \let\glsifplural\@firstoftwo
879 ]%
880 {#1}{#2}%
881 {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
882 }

```

\@glssymbol@

```

883 \def\@glssymbol@#1#2[#3]{%
884 \glstrassignfieldfont{#2}%
885 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
886 }

```

\@GLssymbol@ First letter uppercase version.

```

887 \def\@GLssymbol@#1#2[#3]{%
888 \glstrassignfieldfont{#2}%
889 \@gls@field@link
890 [\let\glscapscase\@secondoftwo]%
891 {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
892 }

```

\@GLSsymbol@ All uppercase version.

```

893 \def\@GLSsymbol@#1#2[#3]{%
894 \glstrassignfieldfont{#2}%
895 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
896 {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
897 }

```

lssymbolplural@ No case-changing version.

```

898 \def\@glssymbolplural@#1#2[#3]{%
899 \glstrassignfieldfont{#2}%
900 \@gls@field@link
901 [\let\glscapscase\@secondoftwo
902 \let\glsifplural\@firstoftwo
903 ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
904 }

```

lssymbolplural@ First letter uppercase version.

```

905 \def\@Glssymbolplural@#1#2[#3]{%

```

```

906 \glstrassignfieldfont{#2}%
907 \@gls@field@link
908 [\let\glscapscase\@secondoftwo
909 \let\glsifplural\@firstoftwo
910 ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
911 }

```

Lsymbolplural@ All uppercase version.

```

912 \def\@GLSsymbol@#1#2[#3]{%
913 \glstrassignfieldfont{#2}%
914 \@gls@field@link
915 [\let\glscapscase\@thirdoftwo
916 \let\glsifplural\@firstoftwo
917 ]%
918 {#1}{#2}%
919 {\@gls@field@font{\Glsaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
920 }

```

\@Glsuseri@ First letter uppercase version.

```

921 \def\@Glsuseri@#1#2[#3]{%
922 \glstrassignfieldfont{#2}%
923 \@gls@field@link
924 [\let\glscapscase\@secondoftwo]{#1}{#2}%
925 {\@gls@field@font{\Glsentryuseri{#2}#3}}%
926 }

```

\@GLSuseri@ All uppercase version.

```

927 \def\@GLSuseri@#1#2[#3]{%
928 \glstrassignfieldfont{#2}%
929 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
930 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}}%
931 }

```

\@Glsuserii@ First letter uppercase version.

```

932 \def\@Glsuserii@#1#2[#3]{%
933 \glstrassignfieldfont{#2}%
934 \@gls@field@link
935 [\let\glscapscase\@secondoftwo]%
936 {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
937 }

```

\@GLSuserii@ All uppercase version.

```

938 \def\@GLSuserii@#1#2[#3]{%
939 \glstrassignfieldfont{#2}%
940 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
941 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}}%
942 }

```

\@Glsuseriii@ First letter uppercase version.

```
943 \def\@Glsuseriii@#1#2[#3]{%
944   \glstrassignfieldfont{#2}%
945   \@gls@field@link
946   [\let\glscapscase\@secondoftwo]%
947   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
948 }
```

\@GLSuseriii@ All uppercase version.

```
949 \def\@GLSuseriii@#1#2[#3]{%
950   \glstrassignfieldfont{#2}%
951   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
952   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
953 }
```

\@Glsuseriv@ First letter uppercase version.

```
954 \def\@Glsuseriv@#1#2[#3]{%
955   \glstrassignfieldfont{#2}%
956   \@gls@field@link
957   [\let\glscapscase\@secondoftwo]%
958   {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
959 }
```

\@GLSuseriv@ All uppercase version.

```
960 \def\@GLSuseriv@#1#2[#3]{%
961   \glstrassignfieldfont{#2}%
962   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
963   {#1}{#2}%
964   {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
965 }
```

\@Glsuserv@ First letter uppercase version.

```
966 \def\@Glsuserv@#1#2[#3]{%
967   \glstrassignfieldfont{#2}%
968   \@gls@field@link
969   [\let\glscapscase\@secondoftwo]%
970   {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
971 }
```

\@GLSuserv@ All uppercase version.

```
972 \def\@GLSuserv@#1#2[#3]{%
973   \glstrassignfieldfont{#2}%
974   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
975   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
976 }
```

\@Glsuservi@ First letter uppercase version.

```
977 \def\@Glsuservi@#1#2[#3]{%
```



```

978 \glstrassignfieldfont{#2}%
979 \@gls@field@link
980 [\let\glscapscase\@secondoftwo]%
981 {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}%
982 }

```

\@GLSuservi@ All uppercase version.

```

983 \def\@GLSuservi@#1#2[#3]{%
984 \glstrassignfieldfont{#2}%
985 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
986 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\Glsentryuservi{#2}#3}}}%
987 }

```

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

\@acrshort No case change.

```

988 \def\@acrshort#1#2[#3]{%
989 \glsdoifexists{#2}%
990 {%
991 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
992 \let\glstrifwasfirstuse\@secondoftwo
993 \let\glsifplural\@secondoftwo
994 \let\glscapscase\@firstofthree
995 \let\glsinsert\@empty
996 \def\glscustomtext{%
997 \acronymfont{\Glsaccesssshort{#2}}#3%
998 }%
999 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1000 }%
1001 \glspostlinkhook
1002 }

```

\@Acrshort First letter uppercase.

```

1003 \def\@Acrshort#1#2[#3]{%
1004 \glsdoifexists{#2}%
1005 {%
1006 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1007 \let\glstrifwasfirstuse\@secondoftwo
1008 \let\glsifplural\@secondoftwo
1009 \let\glscapscase\@secondofthree
1010 \let\glsinsert\@empty
1011 \def\glscustomtext{%
1012 \Glsaccesssshort{#2}}#3%
1013 }%
1014 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1015 }%
1016 \glspostlinkhook
1017 }

```

\@ACRshort All uppercase.

```
1018 \def\@ACRshort#1#2[#3]{%
1019   \glsdoifexists{#2}%
1020   {%
1021     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1022     \let\glxtrifwasfirstuse\@secondoftwo
1023     \let\gl@sifplural\@secondoftwo
1024     \let\glscapscase\@thirdofthree
1025     \let\gl$insert\@empty
1026     \def\glscustomtext{%
1027       \mfirstucMakeUppercase{\acronymfont{\gl@saccesssshort{#2}}#3}%
1028     }%
1029     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1030   }%
1031   \glspostlinkhook
1032 }
```

\@acrshortpl No case change.

```
1033 \def\@acrshortpl#1#2[#3]{%
1034   \glsdoifexists{#2}%
1035   {%
1036     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1037     \let\glxtrifwasfirstuse\@secondoftwo
1038     \let\gl@sifplural\@firstoftwo
1039     \let\glscapscase\@firstofthree
1040     \let\gl$insert\@empty
1041     \def\glscustomtext{%
1042       \acronymfont{\gl@saccesssshortpl{#2}}#3%
1043     }%
1044     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1045   }%
1046   \glspostlinkhook
1047 }
```

\@Acrshortpl First letter uppercase.

```
1048 \def\@Acrshortpl#1#2[#3]{%
1049   \glsdoifexists{#2}%
1050   {%
1051     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1052     \let\glxtrifwasfirstuse\@secondoftwo
1053     \let\gl@sifplural\@firstoftwo
1054     \let\glscapscase\@secondofthree
1055     \let\gl$insert\@empty
1056     \def\glscustomtext{%
1057       \acronymfont{\Glsaccesssshortpl{#2}}#3%
1058     }%
1059     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1060   }%
1061   \glspostlinkhook
```

1062 }

\@ACRshortpl All uppercase.

```
1063 \def\@ACRshortpl#1#2[#3]{%
1064   \glsdoifexists{#2}%
1065   {%
1066     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1067     \let\glxtrifwasfirstuse\@secondoftwo
1068     \let\glsifplural\@firstoftwo
1069     \let\glscapscase\@thirdofthree
1070     \let\glsinsert\@empty
1071     \def\glscustomtext{%
1072       \mfirstucMakeUppercase{\acronymfont{\glsaccessshortpl{#2}}#3}%
1073     }%
1074     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1075   }%
1076   \glspostlinkhook
1077 }
```

\@acrlong No case change.

```
1078 \def\@acrlong#1#2[#3]{%
1079   \glsdoifexists{#2}%
1080   {%
1081     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1082     \let\glxtrifwasfirstuse\@secondoftwo
1083     \let\glsifplural\@secondoftwo
1084     \let\glscapscase\@firstofthree
1085     \let\glsinsert\@empty
1086     \def\glscustomtext{%
1087       \acronymfont{\glsaccesslong{#2}}#3%
1088     }%
1089     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1090   }%
1091   \glspostlinkhook
1092 }
```

\@Acrlong First letter uppercase.

```
1093 \def\@Acrlong#1#2[#3]{%
1094   \glsdoifexists{#2}%
1095   {%
1096     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1097     \let\glxtrifwasfirstuse\@secondoftwo
1098     \let\glsifplural\@secondoftwo
1099     \let\glscapscase\@secondofthree
1100     \let\glsinsert\@empty
1101     \def\glscustomtext{%
1102       \acronymfont{\Glsaccesslong{#2}}#3%
1103     }%
1104     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1105   }
```

```

1105 }%
1106 \glspostlinkhook
1107 }

```

\@ACRlong All uppercase.

```

1108 \def\@ACRlong#1#2[#3]{%
1109   \glsdoidexists{#2}%
1110   {%
1111     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1112     \let\glxtrifwasfirstuse\@secondoftwo
1113     \let\gl@sifplural\@secondoftwo
1114     \let\glscapscase\@thirdofthree
1115     \let\gl$insert\@empty
1116     \def\glscustomtext{%
1117       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslong{#2}}#3}%
1118     }%
1119     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1120   }%
1121   \glspostlinkhook
1122 }

```

\@acrlongpl No case change.

```

1123 \def\@acrlongpl#1#2[#3]{%
1124   \glsdoidexists{#2}%
1125   {%
1126     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1127     \let\glxtrifwasfirstuse\@secondoftwo
1128     \let\gl@sifplural\@firstoftwo
1129     \let\glscapscase\@firstofthree
1130     \let\gl$insert\@empty
1131     \def\glscustomtext{%
1132       \acronymfont{\gl@saccesslongpl{#2}}#3%
1133     }%
1134     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1135   }%
1136   \glspostlinkhook
1137 }

```

\@Acrlongpl First letter uppercase.

```

1138 \def\@Acrlongpl#1#2[#3]{%
1139   \glsdoidexists{#2}%
1140   {%
1141     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1142     \let\glxtrifwasfirstuse\@secondoftwo
1143     \let\gl@sifplural\@firstoftwo
1144     \let\glscapscase\@secondofthree
1145     \let\gl$insert\@empty
1146     \def\glscustomtext{%
1147       \acronymfont{\Glsaccesslongpl{#2}}#3%

```

```

1148 }%
1149 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1150 }%
1151 \glspostlinkhook
1152 }

```

\@ACRlongpl All uppercase.

```

1153 \def\@ACRlongpl#1#2[#3]{%
1154   \glsdoifexists{#2}%
1155   {%
1156     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1157     \let\glstrifwasfirstuse\@secondoftwo
1158     \let\glsifplural\@firstoftwo
1159     \let\glscapscase\@thirdofthree
1160     \let\glsinsert\@empty
1161     \def\glscustomtext{%
1162       \mfirstucMakeUppercase{\acronymfont{\glsaccesslongpl{#2}}#3}%
1163     }%
1164     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1165   }%
1166   \glspostlinkhook
1167 }

```

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

\@glsaddkey

```

1168 \renewcommand*{\@glsaddkey}[7]{%
1169   \key@ifundefined{glossentry}{#1}%
1170   {%
1171     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1172     \appto\@gls@keymap{,{#1}{#1}}%
1173     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
1174     \appto\@newglossaryentryposthook{%
1175       \letcs{\@glo@tmp}{@glo@#1}%
1176       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1177     }%
1178     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1179     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

1180   \ifcsdef{@gls@user@#1@}%
1181   {%
1182     \PackageError{glossaries}%
1183     {Can't define '\string#5' as helper command
1184     '\expandafter\string\csname @gls@user@#1@\endcsname' already
1185     exists}%
1186   }%
1187 }%
1188 {%

```

```

1189 \expandafter\newcommand\expandafter*\expandafter
1190 {\csname @gls@user@#1\endcsname}[2][\%
1191 \new@ifnextchar[%
1192 {\csuse{@gls@user@#1@}{##1}{##2}}}%
1193 {\csuse{@gls@user@#1@}{##1}{##2}[]}}}%
1194 \csdef{@gls@user@#1@}##1##2[##3]{%
1195 \@gls@field@link{##1}{##2}{#3{##2}##3}%
1196 }%
1197 \newrobustcmd*{#5}{%
1198 \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
1199 }%

```

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

```

1200 \ifcsdef{@Gls@user@#1@}%
1201 {%
1202 \PackageError{glossaries}%
1203 {Can't define '\string#6' as helper command
1204 '\expandafter\string\csname @Gls@user@#1\endcsname' already
1205 exists}%
1206 }%
1207 }%
1208 {%
1209 \expandafter\newcommand\expandafter*\expandafter
1210 {\csname @Gls@user@#1\endcsname}[2][\%
1211 \new@ifnextchar[%
1212 {\csuse{@Gls@user@#1@}{##1}{##2}}}%
1213 {\csuse{@Gls@user@#1@}{##1}{##2}[]}}}%
1214 \csdef{@Gls@user@#1@}##1##2[##3]{%
1215 \@gls@field@link[\let\glscaps@case\@secondofthree]%
1216 {##1}{##2}{#4{##2}##3}%
1217 }%
1218 \newrobustcmd*{#6}{%
1219 \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
1220 }%

```

Finally the all caps version (modified):

```

1221 \ifcsdef{@GLS@user@#1@}%
1222 {%
1223 \PackageError{glossaries}%
1224 {Can't define '\string#7' as helper command
1225 '\expandafter\string\csname @GLS@user@#1\endcsname' already
1226 exists}%
1227 }%
1228 }%
1229 {%
1230 \expandafter\newcommand\expandafter*\expandafter
1231 {\csname @GLS@user@#1\endcsname}[2][\%
1232 \new@ifnextchar[%
1233 {\csuse{@GLS@user@#1@}{##1}{##2}}}%
1234 {\csuse{@GLS@user@#1@}{##1}{##2}[]}}}%

```

```

1235 \csdef{@GLS@user@#1@}##1##2[##3]{%
1236 \@gls@field@link[\let\glscapscase\@thirdofthree]%
1237 {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}}%
1238 }%
1239 \newrobustcmd*{#7}{%
1240 \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
1241 }%
1242 }%
1243 {%
1244 \PackageError{glossaries-extra}{Key ‘#1’ already exists}{}%
1245 }%
1246 }

```

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

```
1247 \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.

```

1248 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
1249 \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.

```

1250 \ifglsused{\glslabel}%
1251 {\let\glsxtrifwasfirstuse\@secondoftwo}
1252 {\let\glsxtrifwasfirstuse\@firstoftwo}%

```

Store the category label for convenience.

```

1253 \edef\glscategorylabel{\glscategory{\glslabel}}%
1254 \ifglsused{\glslabel}%
1255 {%
1256 \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
1257 {\KV@glslink@hyperfalse}}}%
1258 }%
1259 {%
1260 \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
1261 {\KV@glslink@hyperfalse}}}%
1262 }%
1263 \glslinkcheckfirsthyperhook
1264 }

```

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.

```

1265 \ifdef\do@gl:disablehyperinlist
1266 {%
1267 \let\@glsxtr@do@gl:disablehyperinlist\do@gl:disablehyperinlist
1268 \renewcommand*{\do@gl:disablehyperinlist}{%

```

```

1269 \glsxtr@do@gl:disablehyperinlist
1270 \glsifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}%
1271 }
1272 }
1273 {}

```

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

```

1274 \define@boolkey{glslink}{noindex}[true]{}
1275 \KV@glslink@noindexfalse

```

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

`\@glslink@opts`

```

1276 \ifdef\@gls@setdefault@glslink@opts
1277 {
1278 \renewcommand*{\@gls@setdefault@glslink@opts}{%
1279 \KV@glslink@noindexfalse
1280 }
1281 }
1282 {

```

Not defined so prepend it to `\do@gl:disablehyperinlist` to achieve the same effect.

```

1283 \newcommand*{\@gls@setdefault@glslink@opts}{%
1284 \KV@glslink@noindexfalse
1285 }
1286 \preto\do@gl:disablehyperinlist{\@gls@setdefault@glslink@opts}
1287 }

```

`\@DefaultGlsOpts` Set the default options for `\glslink` etc.

```

1288 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
1289 \renewcommand*{\@gls@setdefault@glslink@opts}{\setkeys{glslink}{#1}}%
1290 }

```

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

```

1291 \newcommand*{\glstrifindexing}[2]{%
1292 \ifKV@glslink@noindex #2\else #1\fi
1293 }

```

`\glswriteentry` Redefine to test for `indexonlyfirst` category attribute.

```

1294 \renewcommand*{\glswriteentry}[2]{%
1295 \glstrifindexing
1296 {%
1297 \ifgl:indexonlyfirst
1298 \ifgl:sused{#1}
1299 {\gl:xtrdoautoindexname{#1}{dualindex}}%
1300 {#2}%
1301 \else
1302 \glsifattribute{#1}{indexonlyfirst}{true}%
1303 {\ifgl:sused{#1}

```



```

1304      {\glstrdoautoindexname{#1}{dualindex}}}%
1305      {#2}}}%
1306      {#2}}%
1307      \fi
1308    }%
1309    {}%
1310  }

```

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

```

1311 \appto@@do@@wrglossary{\@glstrdo@@wrindex
1312   \glstrdowrglossaryhook{\@gls@label}}%
1313 }

```

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

Similarly for the “noidx” version:

`s@noidxglossary`

```

1314 \appto\gls@noidxglossary{\@glstrdo@@wrindex
1315   \glstrdowrglossaryhook{\@gls@label}}%
1316 }

```

`xtr@do@@wrindex`

```

1317 \newcommand*{\@glstrdo@@wrindex}{%
1318   \glstrdoautoindexname{\@gls@label}{dualindex}}%
1319 }

```

`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.)

```

1320 \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.

```

1321 \newcommand*{\@gls@alt@hyp@opt}[1]{%
1322   \let\glslinkvar\@firstofthree
1323   \let\@gls@hyp@opt@cs#1\relax
1324   \@ifstar{\s@gls@hyp@opt}%
1325   {\@ifnextchar+%
1326     {\@firstoftwo{\p@gls@hyp@opt}}}%
1327   {%
1328     \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
1329     {\@firstoftwo{\@alt@gls@hyp@opt}}}%
1330     {#1}}%
1331   }%
1332 }%
1333 }

```

`alt@glshyp@opt` User version

```

1334 \newcommand*{\@alt@glshyp@opt}[1] [] {%
1335   \let\glslinkvar\@firstofthree
1336   \expandafter\@glshyp@opt@cs\expandafter[\@glshyp@opt@keys,#1]}

```

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

```

1337 \newcommand*{\@glshyp@opt@char}{}

```

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

```

1338 \newcommand*{\@glshyp@opt@keys}{}

```

`rSetAltModifier`

```

1339 \newcommand*{\GlsXtrSetAltModifier}[2] {%
1340   \let\@glshyp@opt\@glshyp@opt
1341   \def\@glshyp@opt@char{#1}%
1342   \def\@glshyp@opt@keys{#2}%
1343 }

```

`\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 `\glsentryshort` and, similarly, the long form commands like `\acrlong` and `\glxtrlong` to use `\glsentrylong`.

```

1344 \renewcommand*{\glsdohyperlink}[2] {%
1345   \hyperlink{#1}{\glxtrprotectlinks#2}}

```

`gl:disablehyper` Redefine in case we have an old version of glossaries.

```

1346 \ifundef\glsdonohyperlink
1347 {%
1348   \renewcommand{\gl:disablehyper}{%
1349     \KV@glslink@hyperfalse
1350     \let\@glslink\glsdonohyperlink
1351     \let\@glstarget\@secondoftwo
1352   }
1353 }
1354 {}

```

`l:sdonohyperlink` 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.

```

1355 \def\glsdonohyperlink#1#2{\glxtrprotectlinks #2}

```

Reset `\@glslink` with patched versions:

```

1356 \ifcsundef{hyperlink}%
1357 {%

```

```

1358 \let\@glslink\glsdonohyperlink
1359 }%
1360 {%
1361 \let\@glslink\glsdohyperlink
1362 }

```

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

```

1363 \newcommand*{\glsxtrprotectlinks}{%
1364 \KV@glslink@hyperfalse
1365 \KV@glslink@noindextrue
1366 \let\@gls@\@glsxtr@p@text@
1367 \let\@Gls@\@Glsxtr@p@text@
1368 \let\@GLS@\@GLSxtr@p@text@
1369 \let\@glspl@\@glsxtr@p@plural@
1370 \let\@Glspl@\@Glsxtr@p@plural@
1371 \let\@GLSpl@\@GLSxtr@p@plural@
1372 \let\@glsxtrshort\@glsxtr@p@short@
1373 \let\@Glsxtrshort\@Glsxtr@p@short@
1374 \let\@GLSxtrshort\@GLSxtr@p@short@
1375 \let\@glsxtrlong\@glsxtr@p@long@
1376 \let\@Glsxtrlong\@Glsxtr@p@long@
1377 \let\@GLSxtrlong\@GLSxtr@p@long@
1378 \let\@glsxtrshortpl\@glsxtr@p@shortpl@
1379 \let\@Glsxtrshortpl\@Glsxtr@p@shortpl@
1380 \let\@GLSxtrshortpl\@GLSxtr@p@shortpl@
1381 \let\@glsxtrlongpl\@glsxtr@p@longpl@
1382 \let\@Glsxtrlongpl\@Glsxtr@p@longpl@
1383 \let\@GLSxtrlongpl\@GLSxtr@p@longpl@
1384 \let\@acrshort\@glsxtr@p@acrshort@
1385 \let\@Acrshort\@Glsxtr@p@acrshort@
1386 \let\@ACRshort\@GLSxtr@p@acrshort@
1387 \let\@acrshortpl\@glsxtr@p@acrshortpl@
1388 \let\@Acrshortpl\@Glsxtr@p@acrshortpl@
1389 \let\@ACRshortpl\@GLSxtr@p@acrshortpl@
1390 \let\@acrlong\@glsxtr@p@acrlong@
1391 \let\@Acrlong\@Glsxtr@p@acrlong@
1392 \let\@ACRlong\@GLSxtr@p@acrlong@
1393 \let\@acrlongpl\@glsxtr@p@acrlongpl@
1394 \let\@Acrlongpl\@Glsxtr@p@acrlongpl@
1395 \let\@ACRlongpl\@GLSxtr@p@acrlongpl@
1396 }

```

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

`@glsxtr@p@text@`

```

1397 \def\@glsxtr@p@text@#1#2[#3]{\@gls{text@{#1}{#2}[#3]}}

```

`@Glsxtr@p@text@`

```

1398 \def\@Glsxtr@p@text@#1#2[#3]{\@Gls{text@{#1}{#2}[#3]}}

```

```

@GLSxtr@p@text@
1399 \def\@GLSxtr@p@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}}

lsxtr@p@plural@
1400 \def\@glsxtr@p@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}}

lsxtr@p@plural@
1401 \def\@Glsxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}}

LSxtr@p@plural@
1402 \def\@GLSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}}

glsxtr@p@short@
1403 \def\@glsxtr@p@short@#1#2[#3]{%
1404 {%
1405   \glsssetabbrvfmt{\glscategory{#2}}}%
1406   \glsabbrvfont{\glsentryshort{#2}}#3}%
1407 }%
1408 }

Glsxtr@p@short@
1409 \def\@Glsxtr@p@short@#1#2[#3]{%
1410 {%
1411   \glsssetabbrvfmt{\glscategory{#2}}}%
1412   \glsabbrvfont{\Glsentryshort{#2}}#3}%
1413 }%
1414 }

GLSxtr@p@short@
1415 \def\@GLSxtr@p@short@#1#2[#3]{%
1416 {%
1417   \glsssetabbrvfmt{\glscategory{#2}}}%
1418   \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshort{#2}}#3}%
1419   }%
1420 }

sxtr@p@shortpl@
1421 \def\@glsxtr@p@shortpl@#1#2[#3]{%
1422 {%
1423   \glsssetabbrvfmt{\glscategory{#2}}}%
1424   \glsabbrvfont{\glsentryshortpl{#2}}#3}%
1425 }%
1426 }

sxtr@p@shortpl@
1427 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
1428 {%
1429   \glsssetabbrvfmt{\glscategory{#2}}}%

```

```

1430 \glsabbrvfont{\Glsentryshortpl{#2}}#3%
1431 }%
1432 }

```

Sxtr@p@shortpl@

```

1433 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
1434   {%
1435     \glssetabbrvfmt{\glscategory{#2}}%
1436     \mfirstucMakeUppercase{\glsabbrvfont{\Glsentryshortpl{#2}}#3}%
1437   }%
1438 }

```

@glsxtr@p@long@

```

1439 \def\@glsxtr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}

```

@Glsxtr@p@long@

```

1440 \def\@Glsxtr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}

```

@GLSxtr@p@long@

```

1441 \def\@GLSxtr@p@long@#1#2[#3]{%
1442   {\mfirstucMakeUppercase{\glslongfont{\Glsentrylong{#2}}#3}}

```

lsxtr@p@longpl@

```

1443 \def\@glsxtr@p@longpl@#1#2[#3]{\Glsentrylongpl{#2}#3}

```

lsxtr@p@longpl@

```

1444 \def\@Glsxtr@p@longpl@#1#2[#3]{\glslongfont{\Glsentrylongpl{#2}}#3}

```

LSxtr@p@longpl@

```

1445 \def\@GLSxtr@p@longpl@#1#2[#3]{%
1446   {\mfirstucMakeUppercase{\glslongfont{\Glsentrylongpl{#2}}#3}}

```

xtr@p@acrshort@

```

1447 \def\@glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}

```

xtr@p@acrshort@

```

1448 \def\@Glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}

```

xtr@p@acrshort@

```

1449 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
1450   {\mfirstucMakeUppercase{\acronymfont{\Glsentryshort{#2}}#3}}

```

r@p@acrshortpl@

```

1451 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}

```

r@p@acrshortpl@

```

1452 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}

```

```

r@p@acrshortpl@
1453 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
1454   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}#3}}}}

```

```

sxtr@p@acrlong@
1455 \def\@glsxtr@p@acrlong@#1#2[#3]{\glsentrylong{#2}#3}}

```

```

sxtr@p@acrlong@
1456 \def\@Glsxtr@p@acrlong@#1#2[#3]{\Glsentrylong{#2}#3}}

```

```

Sxtr@p@acrlong@
1457 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
1458   {\mfirstucMakeUppercase{\glsentrylong{#2}#3}}}}

```

```

tr@p@acrlongpl@
1459 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\glsentrylongpl{#2}#3}}

```

```

tr@p@acrlongpl@
1460 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\Glsentrylongpl{#2}#3}}

```

```

tr@p@acrlongpl@
1461 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
1462   {\mfirstucMakeUppercase{\glsentrylongpl{#2}#3}}}}

```

Commands to minimise conflict.

```

\@glsxtrp@opt
1463 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

```

```

\glsxtrsetpopts  Used in glossary to switch hyperlinks on for the \glsxtrp type of commands.
1464 \newcommand*{\glsxtrsetpopts}[1]{%
1465   \renewcommand*{\@glsxtrp@opt}{#1}%
1466 }

```

```

\lossxtrsetpopts  Used in glossary to switch hyperlinks on for the \glsxtrp type of commands.
1467 \newcommand*{\lossxtrsetpopts}{%
1468   \glsxtrsetpopts{noindex}%
1469 }

```

```

\@@glsxtrp
1470 \newrobustcmd*{\@@glsxtrp}[2]{%
  Add scope.
1471   {%
1472     \let\glspostlinkhook\relax
1473     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
1474   }%
1475 }

```

\@glsxtrp

```
1476 \newrobustcmd*{\@glsxtrp}[2]{%
1477   \ifcsdef{gls#1}%
1478   {%
1479     \@glsxtrp{gls#1}{#2}%
1480   }%
1481   {%
1482     \ifcsdef{glsxtr#1}%
1483     {%
1484       \@glsxtrp{glsxtr#1}{#2}%
1485     }%
1486     {%
1487       \PackageError{glossaries-extra}{‘#1’ not recognised by
1488         \string\glsxtrp}{}%
1489     }%
1490   }%
1491 }
```

\@Glsxtrp

```
1492 \newrobustcmd*{\@Glsxtrp}[2]{%
1493   \ifcsdef{Gls#1}%
1494   {%
1495     \@glsxtrp{Gls#1}{#2}%
1496   }%
1497   {%
1498     \ifcsdef{Glsxtr#1}%
1499     {%
1500       \@glsxtrp{Glsxtr#1}{#2}%
1501     }%
1502     {%
1503       \PackageError{glossaries-extra}{‘#1’ not recognised by
1504         \string\Glsxtrp}{}%
1505     }%
1506   }%
1507 }
```

\@GLSxtrp

```
1508 \newrobustcmd*{\@GLSxtrp}[2]{%
1509   \ifcsdef{GLS#1}%
1510   {%
1511     \@glsxtrp{GLS#1}{#2}%
1512   }%
1513   {%
1514     \ifcsdef{GLSxtr#1}%
1515     {%
1516       \@glsxtrp{GLSxtr#1}{#2}%
1517     }%
1518     {%
1519       \PackageError{glossaries-extra}{‘#1’ not recognised by
```

```

1520      \string\GLSxtrp}{}%
1521    }%
1522  }%
1523 }

```

\glsxtr@entry@p

```

1524 \newrobustcmd*{\glsxtr@headentry@p}[2]{%
1525   \glsifattribute{#1}{headuc}{true}%
1526   {%
1527     \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
1528   }%
1529   {%
1530     \@gls@entry@field{#1}{#2}%
1531   }%
1532 }

```

\glsxtrp Not robust as it needs to expand somewhat.

```

1533 \ifdef\teorpdfstring
1534 {
1535   \newcommand{\glsxtrp}[2]{%
1536     \protect\NoCaseChange
1537     {%
1538       \protect\teorpdfstring
1539       {%
1540         \protect\glsxtrifinmark
1541         {%
1542           \ifcsdef{glsxtrhead#1}%
1543             {%
1544               {\protect\csuse{glsxtrhead#1}{#2}}%
1545             }%
1546             {%
1547               \glsxtr@headentry@p{#2}{#1}%
1548             }%
1549           }%
1550           {%
1551             \@glsxtrp{#1}{#2}%
1552           }%
1553         }%
1554         {%
1555           \protect\@gls@entry@field{#2}{#1}%
1556         }%
1557       }%
1558     }
1559 }
1560 {
1561   \newcommand{\glsxtrp}[2]{%
1562     \protect\NoCaseChange
1563     {%
1564       \protect\glsxtrifinmark

```



```

1565      {%
1566      \ifcsdef{glxstrhead#1}%
1567      {%
1568      {\protect\csuse{glxstrhead#1}}%
1569      }%
1570      {%
1571      \glxstr@headentry@p{#2}{#1}%
1572      }%
1573      }%
1574      {%
1575      \@glxstrp{#1}{#2}%
1576      }%
1577      }%
1578    }
1579 }

```

Provide short synonyms for the most common option.

`\glsp`

```
1580 \newcommand*{\glsp}{\glxtrp{short}}
```

`\glsp`

```
1581 \newcommand*{\glsp}{\glxtrp{text}}
```

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

```

1582 \ifdef\teorpdfstring
1583 {
1584   \newcommand{\Glsxtrp}[2]{%
1585     \protect\NoCaseChange
1586     {%
1587       \protect\teorpdfstring
1588       {%
1589         \protect\glxstrifinmark
1590         {%
1591           \ifcsdef{Glsxtrhead#1}%
1592           {%
1593             {\protect\csuse{Glsxtrhead#1}{#2}}%
1594             }%
1595             {%
1596               \protect\@Gls@entry@field{#2}{#1}%
1597               }%
1598             }%
1599             {%
1600               \@Glsxtrp{#1}{#2}%
1601               }%
1602             }%
1603             {%
1604               \protect\@Gls@entry@field{#2}{#1}%

```

```

1605     }%
1606 }%
1607 }
1608 }
1609 {
1610   \newcommand{\Glsxtrp}[2]{%
1611     \protect\NoCaseChange
1612     {%
1613       \protect\glsxtrifinmark
1614       {%
1615         \ifcsdef{Glsxtrhead#1}%
1616         {%
1617           {\protect\csuse{Glsxtrhead#1}}%
1618           }%
1619           {%
1620             \protect\@Gls@entry@field{#2}{#1}%
1621             }%
1622           }%
1623           {%
1624             \@Glsxtrp{#1}{#2}%
1625             }%
1626           }%
1627     }
1628 }

```

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

```

1629 \ifdef\texorpdfstring
1630 {
1631   \newcommand{\GLSxtrp}[2]{%
1632     \protect\NoCaseChange
1633     {%
1634       \protect\texorpdfstring
1635       {%
1636         \protect\glsxtrifinmark
1637         {%
1638           \ifcsdef{GLSxtr#1}%
1639           {%
1640             {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
1641             }%
1642             {%
1643               \protect\mfirstucMakeUppercase
1644               {%
1645                 \protect\@gls@entry@field{#2}{#1}%
1646                 }%
1647               }%
1648             }%
1649             {%
1650               \@GLSxtrp{#1}{#2}%
1651             }%

```

```

1652     }%
1653     {%
1654         \protect\@gls@entry@field{#2}{#1}%
1655     }%
1656 }%
1657 }
1658 }
1659 {
1660 \newcommand{\GLSxtrp}[2]{%
1661     \protect\NoCaseChange
1662     {%
1663         \protect\glsxtrifinmark
1664         {%
1665             \ifcsdef{GLSxtr#1}%
1666             {%
1667                 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
1668             }%
1669             {%
1670                 \protect\mfirstucMakeUppercase
1671                 {%
1672                     \protect\@gls@entry@field{#2}{#1}%
1673                 }%
1674             }%
1675         }%
1676     }%
1677     \@GLSxtrp{#1}{#2}%
1678 }%
1679 }%
1680 }
1681 }

```

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.

```

1682 \renewcommand*{\@glsunset}[1]{%
1683     \@glsunset{#1}%
1684     \glsxtrpostunset{#1}%
1685 }%

```

`glsxtrpostunset`

```

1686 \newcommand*{\glsxtrpostunset}[1]{%

```

`\@glslocalunset` Local unset.

```

1687 \renewcommand*{\@glslocalunset}[1]{%
1688   \@@glslocalunset{#1}%
1689   \glsxtrpostlocalunset{#1}%
1690 }%

rpostlocalunset
1691 \newcommand*{\glsxtrpostlocalunset}[1]{%

\@glsreset   Global reset.
1692 \renewcommand*{\@glsreset}[1]{%
1693   \@@glsreset{#1}%
1694   \glsxtrpostreset{#1}%
1695 }%

glsxtrpostreset
1696 \newcommand*{\glsxtrpostreset}[1]{%

\@glslocalreset   Local reset.
1697 \renewcommand*{\@glslocalreset}[1]{%
1698   \@@glslocalreset{#1}%
1699   \glsxtrpostlocalreset{#1}%
1700 }%

rpostlocalreset
1701 \newcommand*{\glsxtrpostlocalreset}[1]{%

leEntryCounting   The first argument is the list of categories and the second argument is the value of the en-
trycount attribute.
1702 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
  Enable entry counting:
1703   \glsenableentrycount
  Redefine \gls etc:
1704   \renewcommand*{\gls}{\cgl}%
1705   \renewcommand*{\Gls}{\cGls}%
1706   \renewcommand*{\glspl}{\cglspl}%
1707   \renewcommand*{\Glspl}{\cGlspl}%
1708   \renewcommand*{\GLS}{\cGLS}%
1709   \renewcommand*{\GLSpl}{\cGLSpl}%
  Set the entrycount attribute:
1710   \@glsxtr@setentrycountunsetattr{#1}{#2}%
  In case this command is used again:
1711   \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
1712   \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
1713     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
1714       can't be used with \string\GlsXtrEnableEntryCounting}%
1715     {Use one or other but not both commands}}%
1716 }

```

countunsetattr

```
1717 \newcommand*{\@glstr@setentrycountunsetattr}[2]{%
1718   \@for\@glstr@cat:=#1\do
1719   {%
1720     \ifdefempty{\@glstr@cat}{}%
1721     {%
1722       \glsssetcategoryattribute{\@glstr@cat}{entrycount}{#2}%
1723     }%
1724   }%
1725 }
```

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

enableentrycount

```
1726 \renewcommand*{\glsenableentrycount}{%
  Enable new fields:
1727   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
  Just in case the user has switched on the docdef option.
1728   \renewcommand*{\gls@defdocnewglossaryentry}{%
1729     \renewcommand*{\newglossaryentry}[2]{%
1730       \PackageError{glossaries}{\string\newglossaryentry\space
1731         may only be used in the preamble when entry counting has
1732         been activated}{If you use \string\glsenableentrycount\space
1733         you must place all entry definitions in the preamble not in
1734         the document environment}%
1735     }%
1736   }%
```

New commands to access new fields:

```
1737 \newcommand*{\glscurrentcount}[1]{%
1738   \ifcsundef{glo@\glscdetoklabel{##1}@currentcount}%
1739   {0}{\@gls@entry@field{##1}{currentcount}}%
1740 }%
1741 \newcommand*{\glsprevcount}[1]{%
1742   \ifcsundef{glo@\glscdetoklabel{##1}@prevcount}%
1743   {0}{\@gls@entry@field{##1}{prevcount}}%
1744 }%
```

Adjust post unset and reset:

```
1745 \let\@glstr@entrycount@org@unset\glstrpostunset
1746 \renewcommand*{\glstrpostunset}[1]{%
1747   \@glstr@entrycount@org@unset{##1}%
1748   \@gls@increment@currentcount{##1}%
1749 }%
1750 \let\@glstr@entrycount@org@localunset\glstrpostlocalunset
1751 \renewcommand*{\glstrpostlocalunset}[1]{%
1752   \@glstr@entrycount@org@localunset{##1}%
1753   \@gls@local@increment@currentcount{##1}%
1754 }%
```

```

1755 \let\@glxtr@entrycount@org@reset\glxtrpostreset
1756 \renewcommand*{\glxtrpostreset}[1]{%
1757   \@glxtr@entrycount@org@reset{##1}%
1758   \csgdef{glo@\glsetoklabel{##1}@currcount}{0}%
1759 }%
1760 \let\@glxtr@entrycount@org@localreset\glxtrpostlocalreset
1761 \renewcommand*{\glxtrpostlocalreset}[1]{%
1762   \@glxtr@entrycount@org@localreset{##1}%
1763   \csdef{glo@\glsetoklabel{##1}@currcount}{0}%
1764 }%

```

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

```

1765 \let\@cgl@\@cgl@
1766 \let\@cgl@pl\@cgl@pl
1767 \let\@cgl@S\@cgl@S
1768 \let\@cgl@spl\@cgl@spl
1769 \let\@cgl@S\@cgl@S
1770 \let\@cgl@spl\@cgl@spl

```

The rest is as the original definition.

```

1771 \AtEndDocument{\@gl@write@entrycounts}%
1772 \renewcommand*{\@gl@entry@count}[2]{%
1773   \csgdef{glo@\glsetoklabel{##1}@prevcount}{##2}%
1774 }%
1775 \let\gl@enable@entrycount\relax
1776 \renewcommand*{\gl@enable@entryunitcount}{%
1777   \PackageError{glossaries-extra}{\string\gl@enable@entryunitcount\space
1778     can't be used with \string\gl@enable@entrycount}%
1779   {Use one or other but not both commands}%
1780 }%
1781 }

```

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

```

1782 \renewcommand*{\@gl@write@entrycounts}{%
1783   \immediate\write\@auxout
1784     {\string\providecommand*{\string\@gl@entry@count}[2]{}}%
1785   \count@=0\relax
1786   \forallgl@sentries{\@gl@sentry}{%
1787     \gl@hasattribute{\@gl@sentry}{entrycount}%
1788     {%
1789       \ifgl@sused{\@gl@sentry}%
1790       {%
1791         \immediate\write\@auxout
1792           {\string\@gl@entry@count{\@gl@sentry}{\gl@sentrycurrcount{\@gl@sentry}}}%
1793       }%
1794     }%
1795     \advance\count@ by \@ne
1796   }%

```

```

1797     {}%
1798 }%
1799 \ifnum\count@=0
1800   \GlossariesExtraWarningNoLine{Entry counting has been enabled
1801     \MessageBreak with \string\glsenableentrycount\space but the
1802     \MessageBreak attribute ‘entrycount’ hasn’t
1803     \MessageBreak been assigned to any of the defined
1804     \MessageBreak entries}%
1805 \fi
1806 }

```

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

```

1807 \newcommand*{\glxtrifcounttrigger}[3]{%
1808   \glshasattribute{#1}{entrycount}%
1809   {%
1810     \ifnum\gl Sentryprevcount{#1}>\gl sgetattribute{#1}{entrycount}\relax
1811     #3%
1812   \else
1813     #2%
1814   \fi
1815 }%
1816 {#3}%
1817 }

```

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

`\@@cgl`s@

```

1818 \def\@@cgl s@#1#2[#3]{%
1819   \glxtrifcounttrigger{#2}%
1820   {%
1821     \cgl sformat{#2}{#3}%
1822     \gl sunset{#2}%
1823   }%
1824   {%
1825     \@gl s@{#1}{#2}[#3]%
1826   }%
1827 }%

```

`\@@cgl`s@

```

1828 \def\@@cgl spl@#1#2[#3]{%
1829   \glxtrifcounttrigger{#2}%
1830   {%
1831     \cgl splformat{#2}{#3}%
1832     \gl sunset{#2}%
1833   }%

```

```

1834  {%
1835    \@glspl@{#1}-{#2}[#3]%
1836  }%
1837 }%

```

\@cGls@

```

1838 \def\@cGls@#1#2[#3]{%
1839   \glxtrifcounttrigger{#2}%
1840   {%
1841     \cGlsformat{#2}{#3}%
1842     \glsunset{#2}%
1843   }%
1844   {%
1845     \@Gls@{#1}-{#2}[#3]%
1846   }%
1847 }%

```

\@cGlspl@

```

1848 \def\@cGlspl@#1#2[#3]{%
1849   \glxtrifcounttrigger{#2}%
1850   {%
1851     \cGlsplformat{#2}{#3}%
1852     \glsunset{#2}%
1853   }%
1854   {%
1855     \@Glspl@{#1}-{#2}[#3]%
1856   }%
1857 }%

```

\@cGLS@

```

1858 \def\@cGLS@#1#2[#3]{%
1859   \glxtrifcounttrigger{#2}%
1860   {%
1861     \cGLSformat{#2}{#3}%
1862     \glsunset{#2}%
1863   }%
1864   {%
1865     \@GLS@{#1}-{#2}[#3]%
1866   }%
1867 }%

```

\@cGLSpl@

```

1868 \def\@cGLSpl@#1#2[#3]{%
1869   \glxtrifcounttrigger{#2}%
1870   {%
1871     \cGLSplformat{#2}{#3}%
1872     \glsunset{#2}%
1873   }%
1874   {%

```



```

1875 \cGLSp1@{#1}{#2}[#3]%
1876 }%
1877 }%
1878 %
1879 % Remove default warnings from \cs{cgl}s etc so that it can be used
1880 % interchangeable with \cs{gl}s etc.
1881 %\begin{macro}{\@cgl@s@}
1882 % \begin{macrocode}
1883 \def\@cgl@s@#1#2[#3]{\@gl@s@{#1}{#2}[#3]}

```

\@cGl@s@

```
1884 \def\@cGl@s@#1#2[#3]{\@Gl@s@{#1}{#2}[#3]}
```

\@cglsp1@

```
1885 \def\@cglsp1@#1#2[#3]{\@glsp1@{#1}{#2}[#3]}
```

\@cGLsp1@

```
1886 \def\@cGLsp1@#1#2[#3]{\@GLsp1@{#1}{#2}[#3]}
```

Add all upper case versions not provided by glossaries.

\cGLS

```
1887 \newrobustcmd*{\cGLS}{\@gl@s@hyp@opt\@cGLS}
```

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

```

1888 \newcommand*{\@cGLS}[2][ ]{%
1889 \new@ifnextchar[{\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2}[ ]}]%
1890 }

```

\@cGLS@

```
1891 \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.

```

1892 \newcommand*{\cGLSformat}[2]{%
1893 \expandafter\mfirstucMakeUppercase\expandafter{\cgl@sformat{#1}{#2}}%
1894 }

```

\cGLSp1

```
1895 \newrobustcmd*{\cGLSp1}{\@gl@s@hyp@opt\@cGLSp1}
```

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

```

1896 \newcommand*{\@cGLSp1}[2][ ]{%
1897 \new@ifnextchar[{\@cGLSp1@{#1}{#2}}{\@cGLSp1@{#1}{#2}[ ]}]%
1898 }

```

\@cGLSp1@

```
1899 \def\@cGLSp1@#1#2[#3]{\@GLSp1@{#1}{#2}[#3]}
```

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

```
1900 \newcommand*{\cGLSplformat}[2]{%
1901   \expandafter\mfirstuc\MakeUppercase\expandafter{\cglsplformat{#1}{#2}}}%
1902 }
```

Modify the trigger formats to check for the regular attribute.

`\cglformat`

```
1903 \renewcommand*{\cglformat}[2]{%
1904   \glsifregular{#1}
1905   {\glsentryfirst{#1}}%
1906   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%2%
1907 }
```

`\cGlsformat`

```
1908 \renewcommand*{\cGlsformat}[2]{%
1909   \glsifregular{#1}
1910   {\Glsentryfirst{#1}}%
1911   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%2%
1912 }
```

`\cglsplformat`

```
1913 \renewcommand*{\cglsplformat}[2]{%
1914   \glsifregular{#1}
1915   {\glsentryfirstplural{#1}}%
1916   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%2%
1917 }
```

`\cGlsplformat`

```
1918 \renewcommand*{\cGlsplformat}[2]{%
1919   \glsifregular{#1}
1920   {\Glsentryfirstplural{#1}}%
1921   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}}%2%
1922 }
```

New code similar to above for unit counting.

`defunitcounters`

```
1923 \newcommand*{\@newglossaryentry@defunitcounters}{%
1924   \edef\@glo@countunit{\csuse{@glxtr@categoryattr@{\@glo@category @unitcount}}}%
1925   \ifdefvoid\@glo@countunit
1926   {}%
1927   {%
1928     \@glxtr@ifunitcounter{\@glo@countunit}%
1929     {}%
1930     {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}}%
1931   }%
1932 }
```

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

```
1933 \newcommand*{\@glxtr@unitcountlist}{}
```

@addunitcounter

```
1934 \newcommand*{\@glxtr@addunitcounter}[1]{%
1935 \listadd{\@glxtr@unitcountlist}{#1}%
1936 \ifcsundef{glxtr@theunit@#1}
1937 {%
1938 \ifcsdef{theH#1}%
1939 {\csdef{glxtr@theunit@#1}{\csuse{theH#1}}}%
1940 {\csdef{glxtr@theunit@#1}{\csuse{the#1}}}%
1941 }%
1942 {}%
1943 }
```

r@ifunitcounter

```
1944 \newcommand*{\@glxtr@ifunitcounter}[3]{%
1945 \xifinlist{#1}{\@glxtr@unitcountlist}{#2}{#3}%
1946 }
```

urrentunitcount

```
1947 \newcommand*\@glxtr@currentunitcount[1]{%
1948 glo@\glsdetoklabel{#1}@currunit@glsggetattribute{#1}{unitcount}.%
1949 \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}%
1950 }
```

eviousunitcount

```
1951 \newcommand*\@glxtr@previousunitcount[1]{%
1952 glo@\glsdetoklabel{#1}@prevunit@glsggetattribute{#1}{unitcount}.%
1953 \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}%
1954 }
```

t@currunitcount

```
1955 \newcommand*{\@glxtr@increment@currunitcount}[1]{%
1956 \glshasattribute{#1}{unitcount}%
1957 {%
1958 \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
1959 \ifcsundef{\@glxtr@csname}%
1960 {%
1961 \csgdef{\@glxtr@csname}{1}%
1962 \listcsxadd
1963 {glo@\glsdetoklabel{#1}@unitlist}%
1964 {\glsggetattribute{#1}{unitcount}.%
1965 \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}%
1966 }%
1967 }%
1968 {%
1969 \csxdef{\@glxtr@csname}%
```

```

1970      {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
1971    }%
1972  }%
1973  {}%
1974 }

```

t@currunitcount

```

1975 \newcommand*{\@glx@local@increment@currunitcount}[1]{%
1976   \glshasattribute{#1}{unitcount}%
1977   {%
1978     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
1979     \ifcsundef{\@glxtr@csname}%
1980     {%
1981       \csdef{\@glxtr@csname}{1}%
1982       \listcseadd
1983       {glo@\glsdetoklabel{#1}@unitlist}%
1984       {\glsggetattribute{#1}{unitcount}.%
1985        \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}}%
1986     }%
1987   }%
1988   {%
1989     \csedef{\@glxtr@csname}%
1990     {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
1991   }%
1992 }%
1993 {}%
1994 }

```

r@currunitcount

```

1995 \newcommand*{\@glxtr@currunitcount}[2]{%
1996   \ifcsundef
1997   {glo@\glsdetoklabel{#1}@currunit@#2}%
1998   {0}%
1999   {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}%
2000 }%

```

r@prevunitcount

```

2001 \newcommand*{\@glxtr@prevunitcount}[2]{%
2002   \ifcsundef
2003   {glo@\glsdetoklabel{#1}@prevunit@#2}%
2004   {0}%
2005   {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}%
2006 }%

```

eentryunitcount

```

2007 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
2008   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%

```

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

```

2009 \renewcommand*{\gls@defdocnewglossaryentry}{%
2010 \renewcommand*\newglossaryentry[2]{%
2011 \PackageError{glossaries}{\string\newglossaryentry\space
2012 may only be used in the preamble when entry counting has
2013 been activated}{If you use \string\glsenableentryunitcount\space
2014 you must place all entry definitions in the preamble not in
2015 the document environment}%
2016 }%
2017 }%

```

New commands to access new fields:

```

2018 \newcommand*\glsentrycurrcount[1]{%
2019 \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2020 \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2021 }%
2022 \newcommand*\glsentryprevcount[1]{%
2023 \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2024 \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2025 }%

```

Access total count:

```

2026 \newcommand*\glsentryprevtotalcount[1]{%
2027 \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
2028 {0}%
2029 {%
2030 \number\csuse{glo@\glsdetoklabel{##1}@prevunittotal}
2031 }%
2032 }%

```

Access max value:

```

2033 \newcommand*\glsentryprevmaxcount[1]{%
2034 \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
2035 {0}%
2036 {%
2037 \number\csuse{glo@\glsdetoklabel{##1}@prevunitmax}
2038 }%
2039 }%

```

Adjust post unset and reset:

```

2040 \let\@glsxtr@entryunitcount@org@unset\glsxtrpostunset
2041 \renewcommand*\glsxtrpostunset[1]{%
2042 \@glsxtr@entryunitcount@org@unset{##1}%
2043 \@gls@increment@currunitcount{##1}%
2044 }%
2045 \let\@glsxtr@entryunitcount@org@localunset\glsxtrpostlocalunset
2046 \renewcommand*\glsxtrpostlocalunset[1]{%
2047 \@glsxtr@entryunitcount@org@localunset{##1}%
2048 \@gls@local@increment@currunitcount{##1}%
2049 }%
2050 \let\@glsxtr@entryunitcount@org@reset\glsxtrpostreset

```

```

2051 \renewcommand*{\glstrpostreset}[1]{%
2052   \glshasattribute{##1}{unitcount}%
2053   {%
2054     \edef\@glstr@csname{\@glstr@currentunitcount{##1}}%
2055     \ifcsundef{\@glstr@csname}%
2056     {}%
2057     {\csgdef{\@glstr@csname}{0}}%
2058   }%
2059   {}%
2060 }%
2061 \let\@glstr@entryunitcount@org@localreset\glstrpostlocalreset
2062 \renewcommand*{\glstrpostlocalreset}[1]{%
2063   \@glstr@entryunitcount@org@localreset{##1}%
2064   \glshasattribute{##1}{unitcount}%
2065   {%
2066     \edef\@glstr@csname{\@glstr@currentunitcount{##1}}%
2067     \ifcsundef{\@glstr@csname}%
2068     {}%
2069     {\csdef{\@glstr@csname}{0}}%
2070   }%
2071   {}%
2072 }%

```

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

```

2073 \let\@cgl@\@cgl@
2074 \let\@cgl@pl\@cgl@pl
2075 \let\@cgl@l\@cgl@l
2076 \let\@cgl@pl\@cgl@pl
2077 \let\@cgl@l\@cgl@l
2078 \let\@cgl@pl\@cgl@pl

```

Write information to the aux file.

```

2079 \AtEndDocument{\@gl@write@entryunitcounts}%
2080 \renewcommand*{\@gl@entry@unitcount}[3]{%
2081   \csgdef{glo@gl@detoklabel{##1}@prevunit@##3}{##2}%
2082   \ifcsundef{glo@gl@detoklabel{##1}@prevunittotal}%
2083   {\csgdef{glo@gl@detoklabel{##1}@prevunittotal}{##2}}%
2084   {%
2085     \csxdef{glo@gl@detoklabel{##1}@prevunittotal}{
2086       \number\numexpr\csuse{glo@gl@detoklabel{##1}@prevunittotal}+##2}%
2087     }%
2088     \ifcsundef{glo@gl@detoklabel{##1}@prevunitmax}%
2089     {\csgdef{glo@gl@detoklabel{##1}@prevunitmax}{##2}}%
2090     {%
2091       \ifnum\csuse{glo@gl@detoklabel{##1}@prevunitmax}<##2
2092       \csgdef{glo@gl@detoklabel{##1}@prevunitmax}{##2}%
2093       \fi
2094     }%
2095   }%

```

```

2096 \let\glsenableentryunitcount\relax
2097 \newcommand*{\glsenableentrycount}{%
2098   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
2099     can't be used with \string\glsenableentryunitcount}%
2100   {Use one or other but not both commands}%
2101 }%
2102 }
2103 \onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

2104 \newcommand*{\@gls@entry@unitcount}[3]{%

```

ryunitcounts@do

```

2105 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
2106   \immediate\write\@auxout
2107   {\string\@gls@entry@unitcount
2108     {\@glsentry}%
2109     {\@glsxtr@currunitcount{\@glsentry}{#1}%
2110     }%
2111     {#1}}%
2112 }

```

entryunitcounts

```

2113 \newcommand*{\@gls@write@entryunitcounts}{%
2114   \immediate\write\@auxout
2115   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
2116   \count@=0\relax
2117   \forallglsentries{\@glsentry}{%
2118     \gls@hasattribute{\@glsentry}{unitcount}%
2119     {%
2120       \ifglsused{\@glsentry}%
2121       {%
2122         \forlistcsloop
2123           {\@gls@write@entryunitcounts@do}%
2124           {glo@\glsdetoklabel{\@glsentry}@unitlist}%
2125       }%
2126     }%
2127     \advance\count@ by \@ne
2128   }%
2129 }%
2130 }%
2131 \ifnum\count@=0
2132   \GlossariesExtraWarningNoLine{Entry counting has been enabled
2133     \MessageBreak with \string\glsenableentryunitcount\space but the
2134     \MessageBreak attribute 'unitcount' hasn't
2135     \MessageBreak been assigned to any of the defined
2136     \MessageBreak entries}%
2137 \fi
2138 }

```

tryUnitCounting The first argument is the list of categories, the second argument is the value of the entrycount attribute and the third is the counter name.

```
2139 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
```

Enable entry counting:

```
2140 \glsenableentryunitcount
```

Redefine \gls etc:

```
2141 \renewcommand*{\gls}{\cglsl}%
```

```
2142 \renewcommand*{\Gls}{\cGls}%
```

```
2143 \renewcommand*{\glspl}{\cglspl}%
```

```
2144 \renewcommand*{\Glspl}{\cGlspl}%
```

```
2145 \renewcommand*{\GLS}{\cGLS}%
```

```
2146 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2147 \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%
```

In case this command is used again:

```
2148 \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
```

```
2149 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

```
2150 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
```

```
2151 can't be used with \string\GlsXtrEnableEntryUnitCounting}%
```

```
2152 {Use one or other but not both commands}}%
```

```
2153 }
```

countunsetattr

```
2154 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
```

```
2155 \@for\@glsxtr@cat:=#1\do
```

```
2156 {%
```

```
2157 \ifdefempty{\@glsxtr@cat}{}}%
```

```
2158 {%
```

```
2159 \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
```

```
2160 \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
```

```
2161 }%
```

```
2162 }%
```

```
2163 }
```

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

```
2164 \renewcommand*{\SetGenericNewAcronym}{%
```



```

2165 \let\@Gls@entryname\@Gls@acentryname
2166 \renewcommand{\newacronym}[4][]{%
2167   \ifdefempty{\@glsacronymlists}%
2168   {%
2169     \def\@glo@type{\acronymtype}%
2170     \setkeys{glossentry}{##1}%
2171     \DeclareAcronymList{\@glo@type}%
2172   }%
2173   {%
2174     \glskeylisttok{##1}%
2175     \glslabeltok{##2}%
2176     \glsshorttok{##3}%
2177     \glslongtok{##4}%
2178     \newacronymhook
2179     \protected@edef\@do@newglossaryentry{%
2180       \noexpand\newglossaryentry{\the\glslabeltok}%
2181       {%
2182         type=\acronymtype,%
2183         name={\expandonce{\acronymentry{##2}}},%
2184         sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
2185         text={\the\glsshorttok},%
2186         short={\the\glsshorttok},%
2187         shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
2188         long={\the\glslongtok},%
2189         longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
2190         category=acronym,%
2191         \GenericAcronymFields,%
2192         \the\glskeylisttok
2193       }%
2194     }%
2195     \@do@newglossaryentry
2196   }%
2197   \renewcommand*\acrfullfmt}[3]{%
2198     \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
2199   \renewcommand*\Acrfullfmt}[3]{%
2200     \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
2201   \renewcommand*\ACRfullfmt}[3]{%
2202     \glslink[##1]{##2}{%
2203       \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
2204   \renewcommand*\acrfullplfmt}[3]{%
2205     \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
2206   \renewcommand*\Acrfullplfmt}[3]{%
2207     \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
2208   \renewcommand*\ACRfullplfmt}[3]{%
2209     \glslink[##1]{##2}{%
2210       \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
2211   \renewcommand*\glsentryfull}[1]{\genacrfullformat{##1}{}}%
2212   \renewcommand*\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
2213   \renewcommand*\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%

```

```

2214 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
2215 }

```

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:

```

2216 \let\@glxtr@org@setacronymstyle\setacronymstyle
2217 \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`.

```

2218 \newcommand*{\MakeAcronymsAbbreviations}{%
2219   \renewcommand*{\newacronym}[4][]{%
2220     \newabbreviation[type=\acronymtype,category=acronym,##1]{##2}{##3}{##4}%
2221   }%
2222   \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
2223   \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%
2224   \renewcommand*{\setacronymstyle}[1]{%
2225     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}
2226       unavailable.
2227       Use \string\setabbreviationstyle\space instead.
2228       The original acronym interface can be restored with
2229       \string\RestoreAcronyms}{}%
2230   }%
2231   \renewcommand*{\newacronymstyle}[1]{%
2232     \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
2233       available unless you restore the original acronym interface with
2234       \string\RestoreAcronyms}%
2235     \@glxtr@org@newacronymstyle{##1}%
2236   }%
2237 }

```

Switch acronyms to abbreviations:

```

2238 \MakeAcronymsAbbreviations

```

RestoreAcronyms Restore acronyms to glossaries interface.

```

2239 \newcommand*{\RestoreAcronyms}{%
2240   \SetGenericNewAcronym
2241   \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
2242   \renewcommand*{\acronymfont}[1]{##1}%
2243   \let\setacronymstyle\@glxtr@org@setacronymstyle
2244   \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.

```

2245 \renewcommand*{\@gls@link@checkfirsthyper}{%
2246   \ifglssused{\glslabel}%

```

```

2247 {\let\glxtrifwasfirstuse\@secondoftwo}
2248 {\let\glxtrifwasfirstuse\@firstoftwo}%
2249 \@glxtr@org@checkfirsthyper
2250 }
2251 \glsssetcategoryattribute{acronym}{regular}{false}%
2252 \setacronymstyle{long-short}%
2253 }

```

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

```

2254 \renewcommand*{\glsacspace}[1]{%
2255   \settowidth{\dimen@}{(\firstacronymfont{\glstryshort{#1}})}%
2256   \ifdim\dimen@<\glsacspacemax~\else\space\fi
2257 }

```

`\glsacspacemax` Value used in the above.

```

2258 \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`

```

2259 \newcommand*{\@glxtr@reg@glosslist}{}

```

Save the original definition of `\makeglossaries`:

```

2260 \let\@glxtr@org@makeglossaries\makeglossaries

```

Redefine `\makeglossaries` to take an optional argument. This should be empty for the usual behaviour (all glossaries need processing with an indexing application) or a comma-separated list of glossary labels indicating those glossaries that should be processed with an indexing application.

`\makeglossaries`

```

2261 \renewcommand*{\makeglossaries}[1] []{%
2262   \ifblank{#1}%
2263   {\@glxtr@org@makeglossaries}%
2264   {%
2265     \edef\@glxtr@reg@glosslist{#1}%
2266     \ifundef{\glswrite}{\newwrite\glswrite}{}%
2267     \protected@write\@auxout{}{\string\providecommand
2268       \string\@glorder[1]}{}
2269     \protected@write\@auxout{}{\string\providecommand
2270       \string\@istfilename[1]}{}
2271     \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
2272     \protected@write\@auxout{}{\string\@glorder{\glorder}}
2273     \write\@auxout{\string\providecommand\string\@gl@reference[3]}{}%

```

Iterate through each supplied glossary type and activate it.

```
2274 \@for\@glo@type:=#1\do{%
2275 \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
2276 }%
```

New glossaries must be created before \makeglossaries:

```
2277 \renewcommand*\newglossary[4][]{%
2278 \PackageError{glossaries}{New glossaries
2279 must be created before \string\makeglossaries}{You need
2280 to move \string\makeglossaries\space after all your
2281 \string\newglossary\space commands}}%
```

Any subsequence instances of this command should have no effect

```
2282 \let\@makeglossary\relax
2283 \let\makeglossary\relax
2284 \let\makeglossaries\relax
```

Disable all commands that have no effect after \makeglossaries

```
2285 \@disable@onlypremakeg
```

Allow see key:

```
2286 \let\gls@checkseeallowed\relax
```

Suppress warning about no \makeglossaries

```
2287 \let\warn@nomakeglossaries\relax
2288 \def\warn@noprntglossary{%
2289 \GlossariesWarningNoLine{No \string\printglossary\space
2290 or \string\printglossaries\space
2291 found.^^J(Remove \string\makeglossaries\space if you don't
2292 want
2293 any glossaries.)^^JThis document will not have a glossary}%
2294 }%
```

Adjust display number list to check for type:

```
2295 \renewcommand*\{glsdisplaynumberlist}[1]{%
2296 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2297 {\@glsxtr@idx@displaynumberlist{##1}}%
2298 {\@glsxtr@noidx@displaynumberlist{##1}}%
2299 }%
```

Adjust entry list:

```
2300 \renewcommand*\{glsentrynumberlist}[1]{%
2301 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2302 {\@glsxtr@idx@entrynumberlist{##1}}%
2303 {\@glsxtr@noidx@entrynumberlist{##1}}%
2304 }%
```

Adjust number list loop

```
2305 \renewcommand*\{glsnumberlistloop}[2]{%
2306 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
2307 {%
2308 \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
```

```

2309         not available for glossary ‘##1’-{}%
2310     }%
2311     {\@glxtr@noidx@numberlistloop{##1}{##2}}%
2312 }%

```

Only sanitize sort for normal indexing glossaries.

```

2313 \renewcommand*\glsprestandardsort}[3]{%
2314     \expandafter\DTLifinlist\expandafter{##2}{\@glxtr@reg@glosslist}%
2315     {%
2316         \glsdosanitizesort
2317     }%
2318     {%
2319         \ifglssanitizesort
2320             \@gls@noidx@sanitizesort
2321         \else
2322             \@gls@noidx@nosanitizesort
2323         \fi
2324     }%
2325 }%

```

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

```

2326 \renewcommand*\new@glossaryentry[2]{%
2327     \PackageError{glossaries-extra}{Glossary entries must be defined
2328         in the preamble\MessageBreak when you use the optional argument
2329         of \string\makeglossaries}{Either move your definitions to the
2330         preamble or don’t use the optional argument of
2331         \string\makeglossaries}%
2332 }%

```

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).

```

2333 \renewcommand*\@printgloss@setsort{%
2334     \renewcommand*\@glo@assign@sortkey{%
2335         \edef\@glo@type{\@glo@type}%
2336         \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
2337         {%
2338             \@glo@no@assign@sortkey
2339         }%
2340         {%
2341             \@glo@assign@sortkey
2342         }%
2343     }%
2344     \def\@glo@sorttype{\@glo@default@sorttype}%
2345 }%

```

Check automake setting:

```

2346 \ifglsautomake
2347     \renewcommand*\@gls@doautomake{%
2348         \@for\@gls@type:=\@glxtr@reg@glosslist\do{%
2349             \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%

```

```

2350     }%
2351     }%
2352   \fi
2353 }%
2354 }

```

Display number list for the regular version:

splaynumberlist

```
2355 \let\@glstr@idx@displaynumberlist\glsdisplaynumberlist
```

Display number list for the “noidx” version:

splaynumberlist

```

2356 \newcommand*{\@glstr@noidx@displaynumberlist}[1]{%
2357   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
2358   \ifdef\@gls@loclist
2359   {%
2360     \def\@gls@noidxloclist@sep{%
2361       \def\@gls@noidxloclist@sep{%
2362         \def\@gls@noidxloclist@sep{%
2363           \glsnumlistsep
2364         }%
2365         \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
2366       }%
2367     }%
2368     \def\@gls@noidxloclist@finalsep{}%
2369     \def\@gls@noidxloclist@prev{}%
2370     \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
2371     \@gls@noidxloclist@finalsep
2372     \@gls@noidxloclist@prev
2373   }%
2374   {%
2375     ??\glsdoifexists{#1}%
2376     {%
2377       \GlossariesWarning{Missing location list for ‘#1’. Either
2378         a rerun is required or you haven’t referenced the entry.}%
2379     }%
2380   }%
2381 }%
2382

```

And for the number list loop:

@numberlistloop

```

2383 \newcommand*{\@glstr@noidx@numberlistloop}[3]{%
2384   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
2385   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
2386   \let\@gls@org@glsseeformat\glsseeformat
2387   \let\glsnoidxdisplayloc#2\relax
2388   \let\glsseeformat#3\relax

```

```

2389 \ifdef\@gls@loclist
2390 {%
2391   \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
2392 }%
2393 {%
2394   ??\glsdoifexists{#1}%
2395   {%
2396     \GlossariesWarning{Missing location list for ‘##1’. Either
2397       a rerun is required or you haven’t referenced the entry.}%
2398   }%
2399 }%
2400 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
2401 \let\glsseeformat\@gls@org@glsseeformat
2402 }%

```

Same for entry number list.

entrynumberlist

```

2403 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
2404   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
2405   \ifdef\@gls@loclist
2406   {%
2407     \glsnoidxloclist{\@gls@loclist}%
2408   }%
2409   {%
2410     ??\glsdoifexists{#1}%
2411     {%
2412       \GlossariesWarning{Missing location list for ‘##1’. Either
2413         a rerun is required or you haven’t referenced the entry.}%
2414     }%
2415   }%
2416 }%

```

entrynumberlist

```

2417 \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

```

2418 \renewcommand{\@print@glossary}{%
2419   \makeatletter
2420   \@input@{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
2421   \IfFileExists{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
2422   {}%
2423   {\glsxtrNoGlossaryWarning{\@glo@type}}%
2424   \ifglxindy
2425     \ifcsundef{@xdy@\@glo@type @language}%
2426     {%
2427       \edef\@do@auxoutstuff{%

```

```

2428 \noexpand\AtEndDocument{%
2429 \noexpand\immediate\noexpand\write\@auxout{%
2430 \string\providecommand\string\@xdylanguage[2]{}}%
2431 \noexpand\immediate\noexpand\write\@auxout{%
2432 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
2433 }%
2434 }%
2435 }%
2436 {%
2437 \edef\@do@auxoutstuff{%
2438 \noexpand\AtEndDocument{%
2439 \noexpand\immediate\noexpand\write\@auxout{%
2440 \string\providecommand\string\@xdylanguage[2]{}}%
2441 \noexpand\immediate\noexpand\write\@auxout{%
2442 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
2443 @language\endcsname}}%
2444 }%
2445 }%
2446 }%
2447 \@do@auxoutstuff
2448 \edef\@do@auxoutstuff{%
2449 \noexpand\AtEndDocument{%
2450 \noexpand\immediate\noexpand\write\@auxout{%
2451 \string\providecommand\string\@gls@codepage[2]{}}%
2452 \noexpand\immediate\noexpand\write\@auxout{%
2453 \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
2454 }%
2455 }%
2456 \@do@auxoutstuff
2457 \fi
2458 \renewcommand*{\@warn@nomakeglossaries}{%
2459 \GlossariesWarningNoLine{\string\makeglossaries\space
2460 hasn't been used,^^Jthe glossaries will not be updated}%
2461 }%
2462 }

```

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

`\GlsWarningHead` Header message.

```

2463 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
2464 This document is incomplete. The external file associated with
2465 the glossary '#1' (which should be called \texttt{#2})
2466 hasn't been created.%
2467 }

```

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

```

2468 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
2469 This has probably happened because there are no entries defined
2470 in this glossary.%
2471 }

```


arningEmptyMain The default “main” glossary is empty.

```
2472 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
2473   If you don't want this glossary,
2474   add \texttt{nomain} to your package option list when you load
2475   \texttt{glossaries-extra.sty}. For example:%
2476 }
```

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

```
2477 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
2478   Did you forget to use \texttt{type=#1} when you defined your
2479   entries? If you tried to load entries into this glossary with
2480   \texttt{\string\loadglsentries} did you remember to use
2481   \texttt{[#1]} as the optional argument? If you did, check that
2482   the definitions in the file you loaded all had the type set
2483   to \texttt{\string\glsdefaulttype}:%
2484 }
```

arningCheckFile Advisory message to check the file contents.

```
2485 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
2486   Check the contents of the file \texttt{#1}. If
2487   it's empty, that means you haven't indexed any of your entries in this
2488   glossary (using commands like \texttt{\string\gls} or
2489   \texttt{\string\glsadd}) so this list can't be generated.
2490   If the file isn't empty, the document build process hasn't been
2491   completed.%
2492 }
```

WarningAutoMake Message when automake option has been used.

```
2493 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
2494   You may need to rerun \LaTeX. If you already have, it may be that
2495   \TeX's shell escape doesn't allow you to run
2496   \ifglxindy xindy\else makeindex\fi. Check the
2497   transcript file \texttt{\jobname.log}. If the shell escape is
2498   disabled, try one of the following:
2499
2500   \begin{itemize}
2501     \item Run the external (Lua) application:
2502
2503       \texttt{makeglossaries-lite.lua \string"\jobname\string"}
2504
2505     \item Run the external (Perl) application:
2506
2507       \texttt{makeglossaries \string"\jobname\string"}
2508   \end{itemize}
2509
2510   Then rerun \LaTeX\ on this document.
2511   \GlossariesExtraWarning{Rerun required to build the
2512   glossary ‘#1’ or check TeX's shell escape allows
2513   you to run \ifglxindy xindy\else makeindex\fi}%

```

2514 }

WarningMisMatch Mismatching \makenoidxglossaries.

```
2515 \newcommand{\GlsXtrNoGlsWarningMisMatch}{%
2516   You need to either replace \texttt{\string\makenoidxglossaries}
2517   with \texttt{\string\makeglossaries} or replace
2518   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
2519   \texttt{\string\printnoidxglossary}
2520   (or \texttt{\string\printnoidxglossaries}) and then rebuild
2521   this document.%
2522 }
```

WarningBuildInfo Build advice.

```
2523 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
2524   Try one of the following:
2525   \begin{itemize}
2526     \item Add \texttt{automake} to your package option list when you load
2527         \texttt{glossaries-extra.sty}. For example:
2528
2529         \texttt{\string\usepackage[automake]%
2530             \glsopenbrace glossaries-extra\glsclosebrace}
2531
2532     \item Run the external (Lua) application:
2533
2534         \texttt{makeglossaries-lite.lua \string"\jobname\string"}
2535
2536     \item Run the external (Perl) application:
2537
2538         \texttt{makeglossaries \string"\jobname\string"}
2539   \end{itemize}
2540
2541   Then rerun \LaTeX\ on this document.%
2542 }
```

oGlsWarningTail Final paragraph.

```
2543 \newcommand{\GlsXtrNoGlsWarningTail}{%
2544   This message will be removed once the problem has been fixed.%
2545 }
```

GlsWarningNoOut No out file created. Build advice.

```
2546 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
2547   The file \texttt{#1} doesn't exist. This most likely means you haven't used
2548   \texttt{\string\makeglossaries} or you have used
2549   \texttt{\string\nofiles}. If this is just a draft version of the
2550   document, you can suppress this message using the
2551   \texttt{nomissingglstext} package option.%
2552 }
```

glossarywarning

```
2553 \newcommand*{\@glstr@defaultnoglossarywarning}[1]{%
2554 \glossarysection[\glossarytoctitle]{\glossarytitle}
2555 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
2556 \par
2557 \glstrifemptyglossary{#1}%
2558 {%
2559 \GlsXtrNoGlsWarningEmptyStart\space
2560 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
2561 \medskip
2562 \noindent\texttt{\string\usepackage[nomain\ifglssacronym ,acronym\fi]%
2563 \glsoopenbrace glossaries-extra\glsclosebrace}
2564 \medskip
2565 }%
2566 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
2567 }%
2568 {%
2569 \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
2570 {%
2571 \GlsXtrNoGlsWarningCheckFile
2572 {\jobname.\csname @glotype@\@glo@type @out\endcsname}
2573
2574 \ifglssautomake
2575
2576 \GlsXtrNoGlsWarningAutoMake{#1}
2577
2578 \else
2579
2580 \ifthenelse{\equal{#1}{main}}%
2581 {%
2582 \GlsXtrNoGlsWarningEmptyMain\par
2583 \medskip
2584 \noindent\texttt{\string\usepackage[nomain]%
2585 \glsoopenbrace glossaries-extra\glsclosebrace}
2586 \medskip
2587 }%
2588 {}%
2589
2590 \ifdefequal\makeglossaries\@no@makeglossaries
2591 {%
2592 \GlsXtrNoGlsWarningMisMatch
2593 }%
2594 {%
2595 \GlsXtrNoGlsWarningBuildInfo
2596 }%
2597 \fi
2598 }%
2599 {%
2600 \GlsXtrNoGlsWarningNoOut
```

```

2601      {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
2602   }%
2603 }%
2604 \par
2605 \GlsXtrNoGlsWarningTail
2606 }

```

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.

```

2607 \@ifpackageloaded{glossaries-accsupp}
2608 {

```

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

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

```

2609   \newcommand*{\glsaccessname}[1]{%
2610     \glsnameaccessdisplay
2611     {%
2612       \glstryname{#1}%
2613     }%
2614     {#1}%
2615   }

```

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

```

2616   \newcommand*{\Glsaccessname}[1]{%
2617     \glsnameaccessdisplay
2618     {%
2619       \Glsstryname{#1}%
2620     }%
2621     {#1}%
2622   }

```

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

```

2623   \newcommand*{\GLSaccessname}[1]{%
2624     \glsnameaccessdisplay
2625     {%
2626       \mfirstucMakeUppercase{\glstryname{#1}}%
2627     }%
2628     {#1}%
2629   }

```

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

```

2630   \newcommand*{\glsaccesstext}[1]{%

```

```

2631 \glstextaccessdisplay
2632 {%
2633 \glstentrytext{#1}%
2634 }%
2635 {#1}%
2636 }

```

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

```

2637 \newcommand*{\Glsaccesstext}[1]{%
2638 \glstextaccessdisplay
2639 {%
2640 \Glsentrytext{#1}%
2641 }%
2642 {#1}%
2643 }

```

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

```

2644 \newcommand*{\GLSaccesstext}[1]{%
2645 \glstextaccessdisplay
2646 {%
2647 \mfirstucMakeUppercase{\glstentrytext{#1}}%
2648 }%
2649 {#1}%
2650 }

```

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

```

2651 \newcommand*{\glsaccessplural}[1]{%
2652 \glspluralaccessdisplay
2653 {%
2654 \glstentryplural{#1}%
2655 }%
2656 {#1}%
2657 }

```

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

```

2658 \newcommand*{\Glsaccessplural}[1]{%
2659 \glspluralaccessdisplay
2660 {%
2661 \Glsentryplural{#1}%
2662 }%
2663 {#1}%
2664 }

```

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

```

2665 \newcommand*{\GLSaccessplural}[1]{%
2666 \glspluralaccessdisplay

```

```

2667   {%
2668       \mfirstucMakeUppercase{\glentryplural{#1}}%
2669   }%
2670   {#1}%
2671 }

```

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

```

2672 \newcommand*{\glsaccessfirst}[1]{%
2673   \glsfirstaccessdisplay
2674   {%
2675       \glentryfirst{#1}%
2676   }%
2677   {#1}%
2678 }

```

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

```

2679 \newcommand*{\Glsaccessfirst}[1]{%
2680   \glsfirstaccessdisplay
2681   {%
2682       \Glentryfirst{#1}%
2683   }%
2684   {#1}%
2685 }

```

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

```

2686 \newcommand*{\GLSaccessfirst}[1]{%
2687   \glsfirstaccessdisplay
2688   {%
2689       \mfirstucMakeUppercase{\glentryfirst{#1}}%
2690   }%
2691   {#1}%
2692 }

```

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

```

2693 \newcommand*{\glsaccessfirstplural}[1]{%
2694   \glsfirstpluralaccessdisplay
2695   {%
2696       \glentryfirstplural{#1}%
2697   }%
2698   {#1}%
2699 }

```

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

```

2700 \newcommand*{\Glsaccessfirstplural}[1]{%
2701   \glsfirstpluralaccessdisplay
2702   {%

```

```

2703     \Glsentryfirstplural{#1}%
2704 }%
2705 {#1}%
2706 }

```

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

```

2707 \newcommand*{\GLSaccessfirstplural}[1]{%
2708   \glfirstpluralaccessdisplay
2709   {%
2710     \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
2711   }%
2712   {#1}%
2713 }

```

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

```

2714 \newcommand*{\glsaccesssymbol}[1]{%
2715   \glssymbolaccessdisplay
2716   {%
2717     \glsentrysymbol{#1}%
2718   }%
2719   {#1}%
2720 }

```

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

```

2721 \newcommand*{\GLSaccesssymbol}[1]{%
2722   \glssymbolaccessdisplay
2723   {%
2724     \Glsentrysymbol{#1}%
2725   }%
2726   {#1}%
2727 }

```

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

```

2728 \newcommand*{\GLSaccesssymbol}[1]{%
2729   \glssymbolaccessdisplay
2730   {%
2731     \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
2732   }%
2733   {#1}%
2734 }

```

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

```

2735 \newcommand*{\glsaccesssymbolplural}[1]{%
2736   \glssymbolpluralaccessdisplay
2737   {%
2738     \glsentrysymbolplural{#1}%
2739   }%

```

```

2740     {#1}%
2741 }

```

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

```

2742 \newcommand*{\Glsaccesssymbolplural}[1]{%
2743   \glssymbolpluralaccessdisplay
2744   {%
2745     \Glsentrysymbolplural{#1}%
2746   }%
2747   {#1}%
2748 }

```

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

```

2749 \newcommand*{\GLSaccesssymbolplural}[1]{%
2750   \glssymbolpluralaccessdisplay
2751   {%
2752     \mfirstucMakeUppercase{\Glsentrysymbolplural{#1}}%
2753   }%
2754   {#1}%
2755 }

```

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

```

2756 \newcommand*{\glsaccessdesc}[1]{%
2757   \glsdescriptionaccessdisplay
2758   {%
2759     \glsentrydesc{#1}%
2760   }%
2761   {#1}%
2762 }

```

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

```

2763 \newcommand*{\Glsaccessdesc}[1]{%
2764   \glsdescriptionaccessdisplay
2765   {%
2766     \Glsentrydesc{#1}%
2767   }%
2768   {#1}%
2769 }

```

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

```

2770 \newcommand*{\GLSaccessdesc}[1]{%
2771   \glsdescriptionaccessdisplay
2772   {%
2773     \mfirstucMakeUppercase{\glsentrydesc{#1}}%
2774   }%
2775   {#1}%
2776 }

```


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

```
2777 \newcommand*{\glsaccessdescplural}[1]{%
2778   \glsdescriptionpluralaccessdisplay
2779   {%
2780     \glsentrydescplural{#1}%
2781   }%
2782   {#1}%
2783 }
```

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

```
2784 \newcommand*{\Glsaccessdescplural}[1]{%
2785   \glsdescriptionpluralaccessdisplay
2786   {%
2787     \Glsentrydescplural{#1}%
2788   }%
2789   {#1}%
2790 }
```

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

```
2791 \newcommand*{\GLSaccessdescplural}[1]{%
2792   \glsdescriptionpluralaccessdisplay
2793   {%
2794     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
2795   }%
2796   {#1}%
2797 }
```

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

```
2798 \newcommand*{\glsaccessshort}[1]{%
2799   \glsshortaccessdisplay
2800   {%
2801     \glsentryshort{#1}%
2802   }%
2803   {#1}%
2804 }
```

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

```
2805 \newcommand*{\Glsaccessshort}[1]{%
2806   \glsshortaccessdisplay
2807   {%
2808     \Glsentryshort{#1}%
2809   }%
2810   {#1}%
2811 }
```

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

```

2812 \newcommand*{\GLSaccessshort}[1]{%
2813   \glsshortaccessdisplay
2814   {%
2815     \mfirstucMakeUppercase{\glsentryshort{#1}}%
2816   }%
2817   {#1}%
2818 }

```

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

```

2819 \newcommand*{\glssaccessshortpl}[1]{%
2820   \glsshortpluralaccessdisplay
2821   {%
2822     \glsentryshortpl{#1}%
2823   }%
2824   {#1}%
2825 }

```

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

```

2826 \newcommand*{\Glsaccessshortpl}[1]{%
2827   \glsshortpluralaccessdisplay
2828   {%
2829     \Glsentryshortpl{#1}%
2830   }%
2831   {#1}%
2832 }

```

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

```

2833 \newcommand*{\GLSaccessshortpl}[1]{%
2834   \glsshortpluralaccessdisplay
2835   {%
2836     \mfirstucMakeUppercase{\glsentryshortpl{#1}}%
2837   }%
2838   {#1}%
2839 }

```

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

```

2840 \newcommand*{\glsaccesslong}[1]{%
2841   \glslongaccessdisplay{\glsentrylong{#1}}{#1}%
2842 }

```

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

```

2843
2844 \newcommand*{\Glsaccesslong}[1]{%
2845   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
2846 }

```

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

```

2847 \newcommand*{\GLSaccesslong}[1]{%
2848   \glslongaccessdisplay
2849   {%
2850     \mfirstucMakeUppercase{\glsentrylong{#1}}%
2851   }%
2852   {#1}%
2853 }

```

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

```

2854 \newcommand*{\glsaccesslongpl}[1]{%
2855   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
2856 }

```

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

```

2857
2858 \newcommand*{\Glsaccesslongpl}[1]{%
2859   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
2860 }

```

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

```

2861 \newcommand*{\GLSaccesslongpl}[1]{%
2862   \glslongpluralaccessdisplay
2863   {%
2864     \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
2865   }%
2866   {#1}%
2867 }

```

End of if part

```

2868 }
2869 {

```

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

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

```

2870 \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.

```

2871 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

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

```

2872 \newcommand*{\GLSaccessname}[1]{%
2873   \protect\mfirstucMakeUppercase{\glsentryname{#1}}%

```

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

```

2874 \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.
2875 `\newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}`

`\GLSaccesstext` Display the text value (no link and no check for existence). converted to upper case.
2876 `\newcommand*{\GLSaccesstext}[1]{%`
2877 `\protect\mfirstucMakeUppercase{\glsentrytext{#1}}}`

`glsaccessplural` Display the plural value (no link and no check for existence).
2878 `\newcommand*{\glsaccessplural}[1]{\glsentryplural{#1}}`

`GLsaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.
2879 `\newcommand*{\GLsaccessplural}[1]{\GLsentryplural{#1}}`

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.
2880 `\newcommand*{\GLSaccessplural}[1]{%`
2881 `\protect\mfirstucMakeUppercase{\glsentryplural{#1}}}`

`\glsaccessfirst` Display the first value (no link and no check for existence).
2882 `\newcommand*{\glsaccessfirst}[1]{\glsentryfirst{#1}}`

`\GLsaccessfirst` Display the first value (no link and no check for existence) with the first letter converted to upper case.
2883 `\newcommand*{\GLsaccessfirst}[1]{\GLsentryfirst{#1}}`

`\GLSaccessfirst` Display the first value (no link and no check for existence). converted to upper case.
2884 `\newcommand*{\GLSaccessfirst}[1]{%`
2885 `\protect\mfirstucMakeUppercase{\glsentryfirst{#1}}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence).
2886 `\newcommand*{\glsaccessfirstplural}[1]{\glsentryfirstplural{#1}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.
2887 `\newcommand*{\GLsaccessfirstplural}[1]{\GLsentryfirstplural{#1}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.
2888 `\newcommand*{\GLSaccessfirstplural}[1]{%`
2889 `\protect\mfirstucMakeUppercase{\glsentryfirstplural{#1}}}`

`glsaccesssymbol` Display the symbol value (no link and no check for existence).
2890 `\newcommand*{\glsaccesssymbol}[1]{\glsentrysymbol{#1}}`

`GLsaccesssymbol` Display the symbol value (no link and no check for existence) with the first letter converted to upper case.
2891 `\newcommand*{\GLsaccesssymbol}[1]{\GLsentrysymbol{#1}}`

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

```

2892 \newcommand*{\GLSaccesssymbol}[1]{%
2893 \protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}
```

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

```

2894 \newcommand*{\glaccesssymbolplural}[1]{\glentrysymbolplural{#1}}
```

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

```

2895 \newcommand*{\GLSaccesssymbolplural}[1]{\GLentrysymbolplural{#1}}
```

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

```

2896 \newcommand*{\GLSaccesssymbolplural}[1]{%
2897 \protect\mfirstucMakeUppercase{\glentrysymbolplural{#1}}}
```

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

```

2898 \newcommand*{\glaccessdesc}[1]{\glentrydesc{#1}}
```

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

```

2899 \newcommand*{\GLsaccessdesc}[1]{\GLentrydesc{#1}}
```

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

```

2900 \newcommand*{\GLSAccessdesc}[1]{%
2901 \protect\mfirstucMakeUppercase{\glentrydesc{#1}}}
```

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

```

2902 \newcommand*{\glaccessdescplural}[1]{\glentrydescplural{#1}}
```

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

```

2903 \newcommand*{\GLsaccessdescplural}[1]{\GLentrydescplural{#1}}
```

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

```

2904 \newcommand*{\GLSaccessdescplural}[1]{%
2905 \protect\mfirstucMakeUppercase{\glentrydescplural{#1}}}
```

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

```

2906 \newcommand*{\glaccessshort}[1]{\glentryshort{#1}}
```

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

```

2907 \newcommand*{\GLsaccessshort}[1]{\GLentryshort{#1}}
```

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

```

2908 \newcommand*{\GLSaccessshort}[1]{%
2909 \protect\mfirstucMakeUppercase{\glentryshort{#1}}}

```

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

```

2910 \newcommand*{\lsaccessshortpl}[1]{\glentryshortpl{#1}}

```

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

```

2911 \newcommand*{\lSaccessshortpl}[1]{\Glentryshortpl{#1}}

```

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

```

2912 \newcommand*{\LSaccessshortpl}[1]{%
2913 \protect\mfirstucMakeUppercase{\glentryshortpl{#1}}}

```

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

```

2914 \newcommand*{\glsaccesslong}[1]{\glentrylong{#1}}

```

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

```

2915 \newcommand*{\Glsaccesslong}[1]{\Glentrylong{#1}}

```

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

```

2916 \newcommand*{\GLSaccesslong}[1]{%
2917 \protect\mfirstucMakeUppercase{\glentrylong{#1}}}

```

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

```

2918 \newcommand*{\glsaccesslongpl}[1]{\glentrylongpl{#1}}

```

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

```

2919 \newcommand*{\Glsaccesslongpl}[1]{\Glentrylongpl{#1}}

```

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

```

2920 \newcommand*{\GLSaccesslongpl}[1]{%
2921 \protect\mfirstucMakeUppercase{\glentrylongpl{#1}}}

```

End of else part

```

2922 }

```

1.5 Categories

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

```

2923 \glsaddstoragekey{category}{general}{\glscategory}

```

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

```

2924 \newcommand{\glsifcategory}[4]{%
2925 \ifglsseldeq{#1}{category}{#2}{#3}{#4}%
2926 }

```

Categories can have attributes.

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

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

```
2927 \newcommand*{\glissetcategoryattribute}[3]{%
2928   \csdef{@glxtr@categoryattr@@#1@#2}{#3}%
2929 }
```

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

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

```
2930 \newcommand*{\glsggetcategoryattribute}[2]{%
2931   \csuse{@glxtr@categoryattr@@#1@#2}%
2932 }
```

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

Tests if the category has the given attribute set.

```
2933 \newcommand*{\glshascategoryattribute}[4]{%
2934   \ifcsvoid{@glxtr@categoryattr@@#1@#2}{#4}{#3}%
2935 }
```

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

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

```
2936 \newcommand*{\glissetattribute}[3]{%
2937   \glissetcategoryattribute{\glscategory{#1}}{#2}{#3}%
2938 }
```

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

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

```
2939 \newcommand*{\glsggetattribute}[2]{%
```

```

2940 \glsggetcategoryattribute{\glscategory{#1}}{#2}%
2941 }

```

`\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.

```

2942 \newcommand*\glshasattribute}[4]{%
2943   \ifglentryexists{#1}%
2944   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
2945   {#4}%
2946 }

```

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

True if category has the attribute with the given value.

```

2947 \newcommand{\glusifcategoryattribute}[5]{%
2948   \ifcsundef{@glxtr@categoryattr@#1@#2}%
2949   {#5}%
2950   {\ifcsstring{@glxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
2951 }

```

`\glusifattribute` `\glusifattribute{<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.

```

2952 \newcommand{\glusifattribute}[5]{%
2953   \ifglentryexists{#1}%
2954   {\glusifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
2955   {#5}%
2956 }

```

Set attributes for the default general category:

```

2957 \glissetcategoryattribute{general}{regular}{true}

```

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

```

2958 \glissetcategoryattribute{acronym}{regular}{true}

```

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

```

2959 \newcommand*\glissetregularcategory}[1]{%

```



```

2960 \glssetcategoryattribute{#1}{regular}{true}%
2961 }

```

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

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

```

2962 \newcommand{\glsifregularcategory}[3]{%
2963   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
2964 }

```

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

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

```

2965 \newcommand{\glsifnotregularcategory}[3]{%
2966   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
2967 }

```

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

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

```

2968 \newcommand{\glsifregular}[3]{%
2969   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
2970 }

```

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

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

```

2971 \newcommand{\glsifnotregular}[3]{%
2972   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
2973 }

```

`\glsforeachincategory` `\glsforeachincategory[<glossary labels>]{<category-label>}{<glossary-cs>}{<label-cs>}{<body>}`

Iterates through all entries in all the glossaries (or just those listed in *<glossary labels>*) and does *<body>* if the category matches *<category-label>*. The control sequences *<glossary-cs>* and

`<label-cs>` may be used in `<body>` to access the glossary label and entry label for the current iteration.

```

2974 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
2975   \forallglossaries[#1]{#3}%
2976   {%
2977     \forglentries[#3]{#4}%
2978     {%
2979       \glsifcategory{#4}{#2}{#5}{}%
2980     }%
2981   }%
2982 }

```

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.

```

2983 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
2984   \forallglossaries[#1]{#4}%
2985   {%
2986     \forglentries[#4]{#5}%
2987     {%
2988       \glsifattribute{#5}{#2}{#3}{#6}{}%
2989     }%
2990   }%
2991 }

```

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

```

2992 \ifdef\newterm
2993 {%

```

`\newterm`

```

2994   \renewcommand*{\newterm}[2][ ]{%
2995     \newglossaryentry{#2}%
2996     {type={index},category=index,name={#2},%
2997       description={\glstrpostdescription\nopostdesc},#1}%
2998   }

```

Indexed terms are regular by default.

```

2999   \glsssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

3000   \newcommand*{\glstrpostdescindex}{}

```

```
3001 }
3002 {}
```

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.

```
3003 \ifdef\printsymbols
3004 {%
```

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

```
3005 \newcommand*{\glsxtrnewsymbol}[3] [] {%
3006 \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
3007 }
```

Symbols are regular by default.

```
3008 \glssetcategoryattribute{symbol}{regular}{true}
```

`rpostdescsymbol`

```
3009 \newcommand*{\glsxtrpostdescsymbol}{}

3010 }
3011 {}
```

Similar for the numbers option.

```
3012 \ifdef\printnumbers
3013 {%
```

`glsxtrnewnumber`

```
3014 \ifdef\printnumbers
3015 \newcommand*{\glsxtrnewnumber}[3] [] {%
3016 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
3017 }
```

Numbers are regular by default.

```
3018 \glssetcategoryattribute{number}{regular}{true}
```

`rpostdescnumber`

```
3019 \newcommand*{\glsxtrpostdescnumber}{}

3020 }
3021 {}
```

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

```
3022 \newcommand*{\glsxtrsetcategory}[2] {%
3023 \@for\@glsxtr@label:=#1\do
3024 {%
```

```

3025 \glsfieldxdef{\@glsxtr@label}{category}{#2}%
3026 }%
3027 }

```

`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.

```

3028 \newcommand*{\glsxtrsetcategoryforall}[2]{%
3029 \forallglossaries[#1]{\@glsxtr@type}{%
3030 \forallglsentries[\@glsxtr@type]{\@glsxtr@label}%
3031 {%
3032 \glsfieldxdef{\@glsxtr@label}{category}{#2}%
3033 }%
3034 }%
3035 }

```

`trfieldtitlecase` `\glsxtrfieldtitlecase{<label>}{<field>}`

Apply title casing to the contents of the given field.

```

3036 \newcommand*{\glsxtrfieldtitlecase}[2]{%
3037 \expandafter\glsxtrfieldtitlecasesecs\expandafter
3038 {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
3039 }

```

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

```

3040 \newcommand*{\glsxtrfieldtitlecasesecs}[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.

```

3041 \@ifpackageloaded{glossaries-accsupp}
3042 {
3043 \renewcommand*{\glossentrydesc}[1]{%
3044 \glsdoifexistsorwarn{#1}%
3045 {%
3046 \glssetabbrvfmt{\glscategory{#1}}%

```

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

```

3047 \glshasattribute{#1}{glossdescfont}%
3048 {%
3049 \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
3050 \ifcsdef{\@glsxtr@attrval}%
3051 {%
3052 \letcs{\@glsxtr@glossdescfont}{\@glsxtr@attrval}%

```

```

3053     }%
3054     {%
3055         \GlossariesExtraWarning{Unknown control sequence name
3056         '\@glxtr@attrval' supplied in glossdescfont attribute
3057         for entry '#1'. Ignoring}%
3058         \let\@glxtr@glossdescfont\@firstofone
3059     }%
3060 }%
3061 {\let\@glxtr@glossdescfont\@firstofone}%
3062 \glsifattribute{#1}{glossdesc}{firstuc}%
3063 {%
3064     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
3065 }%
3066 {%
3067     \glsifattribute{#1}{glossdesc}{title}%
3068     {%
3069         \@glxtr@do@titlecaps@warn
3070         \glsdescriptionaccessdisplay
3071         {%
3072             \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
3073         }%
3074         {#1}%
3075     }%
3076     {%
3077         \@glxtr@glossdescfont{\glsaccessdesc{#1}}%
3078     }%
3079 }%
3080 }%
3081 }
3082 }
3083 {
3084     \renewcommand*{\glossentrydesc}[1]{%
3085         \glsdoifexistsorwarn{#1}%
3086         {%
3087             \glssetabbrvfmt{\glscategory{#1}}%
3088             \glsattribute{#1}{glossdescfont}%
3089             {%
3090                 \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
3091                 \ifcsdef{\@glxtr@attrval}%
3092                 {%
3093                     \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
3094                 }%
3095                 {%
3096                     \GlossariesExtraWarning{Unknown control sequence name
3097                     '\@glxtr@attrval' supplied in glossdescfont attribute
3098                     for entry '#1'. Ignoring}%
3099                     \let\@glxtr@glossdescfont\@firstofone
3100                 }%
3101             }%

```

```

3102      {\let\@glxtr@glossdescfont\@firstofone}%
3103      \glsifattribute{#1}{glossdesc}{firstuc}%
3104      {%
3105        \@glxtr@glossdescfont{\Glsentrydesc{#1}}%
3106      }%
3107      {%
3108        \glsifattribute{#1}{glossdesc}{title}%
3109        {%
3110          \@glxtr@do@titlecaps@warn
3111          \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
3112        }%
3113        {%
3114          \@glxtr@glossdescfont{\Glsentrydesc{#1}}%
3115        }%
3116      }%
3117    }%
3118  }
3119 }

```

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

```

3120 \ifpackageloaded{glossaries-accsupp}
3121 {
3122   \renewcommand*{\glossentryname}[1]{%
3123     \@glsdoifexistsorwarn{#1}%
3124     {%
3125       \glsetabbrvfmt{\glscategory{#1}}%

```

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

```

3126   \glshasattribute{#1}{glossnamefont}%
3127   {%
3128     \edef\@glxtr@attrval{\glsetattribute{#1}{glossnamefont}}%
3129     \ifcsdef{\@glxtr@attrval}%
3130     {%
3131       \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
3132     }%
3133     {%
3134       \GlossariesExtraWarning{Unknown control sequence name
3135         ‘\@glxtr@attrval’ supplied in glossnamefont attribute
3136         for entry ‘#1’. Reverting to default \string\glsnamefont}%
3137       \let\@glxtr@glossnamefont\glsnamefont
3138     }%
3139   }%
3140   {\let\@glxtr@glossnamefont\glsnamefont}%
3141   \glsifattribute{#1}{glossname}{firstuc}%
3142   {%
3143     \glsnameaccessdisplay
3144     {%
3145       \@glxtr@glossnamefont{\Glsentryname{#1}}%
3146     }%

```

```

3147     {#1}%
3148 }%
3149 {%
3150   \glsifattribute{#1}{glossname}{title}%
3151   {%
3152     \@glstr@do@titlecaps@warn
3153     \glsnameaccessdisplay
3154     {%
3155       \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}%
3156     }%
3157     {#1}%
3158   }%
3159   {%
3160     \glsifattribute{#1}{glossname}{uc}%
3161     {%
3162       \glsnameaccessdisplay
3163       {%

```

Hide the label from the upper-casing command.

```

3164         \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
3165         \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
3166       }%
3167       {#1}%
3168     }%
3169     {%
3170       \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
3171       \glsnameaccessdisplay
3172       {%
3173         \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
3174       }%
3175       {#1}%
3176     }%
3177   }%
3178 }%

```

Do post-name hook:

```

3179   \glstrpostnamehook{#1}%
3180 }%
3181 }
3182 }
3183 {
3184   \renewcommand*{\glossentryname}[1]{%
3185     \@glsdofexistsorwarn{#1}%
3186     {%
3187       \glsetabbrvfmt{\glscategory{#1}}%
3188       \glshasattribute{#1}{glossnamefont}%
3189     }%
3190     \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
3191     \ifcsdef{\@glstr@attrval}%
3192     {%

```

```

3193     \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
3194 }%
3195 {%
3196     \GlossariesExtraWarning{Unknown control sequence name
3197     '\@glxtr@attrval' supplied in glossnamefont attribute
3198     for entry '#1'. Reverting to default \string\glnamefont}%
3199     \let\@glxtr@glossnamefont\glnamefont
3200 }%
3201 }%
3202 {\let\@glxtr@glossnamefont\glnamefont}%
3203 \glsifattribute{#1}{glossname}{firstuc}%
3204 {%
3205     \@glxtr@glossnamefont{\Glsentryname{#1}}%
3206 }%
3207 {%
3208     \glsifattribute{#1}{glossname}{title}%
3209     {%
3210         \@glxtr@do@titlecaps@warn
3211         \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
3212     }%
3213     {%
3214         \glsifattribute{#1}{glossname}{uc}%
3215         {%

```

Hide the label from the upper-casing command.

```

3216     \letcs{\glo@name}{\glo@\glsetoklabel{#1}@name}%
3217     \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
3218 }%
3219 {%

```

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

```

3220     \letcs{\glo@name}{\glo@\glsetoklabel{#1}@name}%
3221     \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
3222 }%
3223 }%
3224 }%

```

Do post-name hook.

```

3225     \glxtrpostnamehook{#1}%
3226 }%
3227 }
3228 }

```

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

```

3229 \@ifpackageloaded{glossaries-accsupp}
3230 {
3231     \renewcommand*{\Glossentryname}[1]{%
3232         \@glsetifexistsorwarn{#1}%
3233         {%
3234             \glsetabbrvfmt{\glscategory{#1}}%

```


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

```

3235 \glshasattribute{#1}{glossnamefont}%
3236 {%
3237 \edef\@glxstr@attrval{\glsggetattribute{#1}{glossnamefont}}%
3238 \ifcsdef{\@glxstr@attrval}%
3239 {%
3240 \letcs{\@glxstr@glossnamefont}{\@glxstr@attrval}%
3241 }%
3242 {%
3243 \GlossariesExtraWarning{Unknown control sequence name
3244 '\@glxstr@attrval' supplied in glossnamefont attribute
3245 for entry '#1'. Reverting to default \string\glsgnamefont}%
3246 \let\@glxstr@glossnamefont\glsgnamefont
3247 }%
3248 }%
3249 {\let\@glxstr@glossnamefont\glsgnamefont}%
3250 \glsgnameaccessdisplay
3251 {%
3252 \@glxstr@glossnamefont{\Glsentryname{#1}}%
3253 }%
3254 {#1}%

```

Do post-name hook:

```

3255 \glxstrpostnamehook{#1}%
3256 }%
3257 }
3258 }
3259 {
3260 \renewcommand*{\Glossentryname}[1]{%
3261 \@glsgdoifexistsorwarn{#1}%
3262 {%
3263 \glsgsetabbrvfmt{\glsgcategory{#1}}%
3264 \glshasattribute{#1}{glossnamefont}%
3265 {%
3266 \edef\@glxstr@attrval{\glsggetattribute{#1}{glossnamefont}}%
3267 \ifcsdef{\@glxstr@attrval}%
3268 {%
3269 \letcs{\@glxstr@glossnamefont}{\@glxstr@attrval}%
3270 }%
3271 {%
3272 \GlossariesExtraWarning{Unknown control sequence name
3273 '\@glxstr@attrval' supplied in glossnamefont attribute
3274 for entry '#1'. Reverting to default \string\glsgnamefont}%
3275 \let\@glxstr@glossnamefont\glsgnamefont
3276 }%
3277 }%
3278 {\let\@glxstr@glossnamefont\glsgnamefont}%
3279 \@glxstr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

3280     \glxtrpostnamehook{#1}%
3281   }%
3282 }
3283 }

```

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

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

```

3284 \newcommand*{\glxtrpostnamehook}[1]{%
3285   \def\@glsnumberformat{glsnumberformat}%
3286   \glxtrdoautoindexname{#1}{indexname}%

```

Allow categories to hook in here.

```

3287   \csuse{glxtrpostname\glscategory{\glscurrententrylabel}}%
3288 }

```

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

```

3289 \newif\if@glxtr@format@override
3290 \@glxtr@format@overridefalse

```

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

`xFormatOverride`

```

3291 \@ifpackageloaded{hyperref}
3292 {

```

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

```

3293   \ifHy@hyperindex
3294     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
3295       \@glxtr@format@override true
3296       \appto\theindex{\let\glshypernumber\@firstofone}%
3297     }
3298   \else
3299     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
3300       \@glxtr@format@override true
3301       \appto\theindex{\let\glshypernumber\hyperpage}%
3302     }
3303   \fi
3304 }
3305 {
3306   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
3307     \@glxtr@format@override true
3308   }
3309 }
3310 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

doautoindexname

```
3311 \newcommand*{\glstrdoautoindexname}[2]{%
3312   \glshasattribute{#1}{#2}%
3313   {%
```

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.

```
3314   \@glstr@autoindex@setname{#1}%
```

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

```
3315   \protected@edef\@glstr@attrval{\glsggetattribute{#1}{#2}}%
3316   \if@glstr@format@override
3317     \ifdefstring{\@glsnumberformat}{\glsgnumberformat}{}%
3318     {\let\@glstr@attrval\@glsgnumberformat}%
3319   \fi
3320   \ifdefstring{\@glstr@attrval}{true}%
3321   {%
3322     {\eappto\@glo@name{\@glstr@autoindex@encap\@glstr@attrval}}%
3323     \expandafter\index\expandafter{\@glo@name}%
3324   }%
3325   {%
3326 }
```

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

```
3327 \newcommand*{\@glstr@autoindex@setname}[1]{%
3328   \def\@glo@name{\string\glstryname{#1}}%
3329   \glsetentryfield{\@glo@sort}{#1}{sort}%
3330   \@glsgcheckmkidxchars\@glo@sort
3331   \@glstr@autoindex@doextra@esc\@glo@sort
3332   \epreto\@glo@name{\@glo@sort\@glstr@autoindex@at}%
3333 }
```

dex@doextra@esc

```
3334 \newcommand*{\@glstr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
3335   \ifx\@glstr@autoindex@esc\@glsgquotechar
3336   \else
3337     \def\@glsgcheckedmkidx{%
3338       \edef\@glstr@checkspch{%
3339         \noexpand\@glstr@autoindex@escquote\expandonce{#1}%
3340         \noexpand\@empty\@glstr@autoindex@esc\noexpand\@nnil
3341         \@glstr@autoindex@esc\noexpand\@empty\noexpand\@glstr@endescspch}%
3342       \@glstr@checkspch
3343       \let#1\@glsgcheckedmkidx\relax
3344     \fi
```

Escape actual character unless it has already been escaped.

```
3345   \ifx\@glstr@autoindex@at\@glsgactualchar
3346   \else
```

```

3347 \def\@gls@checkedmkidx{}%
3348 \edef\@@glsxtr@checkspch{%
3349     \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
3350     \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
3351     \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
3352 @@glsxtr@checkspch
3353 \let#1\@gls@checkedmkidx\relax
3354 \fi

Escape level character unless it has already been escaped.
3355 \ifx\@glsxtr@autoindex@level\@gls@levelchar
3356 \else
3357 \def\@gls@checkedmkidx{}%
3358 \edef\@@glsxtr@checkspch{%
3359     \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
3360     \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
3361     \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
3362 @@glsxtr@checkspch
3363 \let#1\@gls@checkedmkidx\relax
3364 \fi

Escape encap character unless it has already been escaped.
3365 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
3366 \else
3367 \def\@gls@checkedmkidx{}%
3368 \edef\@@glsxtr@checkspch{%
3369     \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
3370     \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
3371     \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
3372 @@glsxtr@checkspch
3373 \let#1\@gls@checkedmkidx\relax
3374 \fi
3375 }

```

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.

```
3376 \newcommand*{\@glsxtr@autoindex@at}{}
```

trSetActualChar Set the actual character.

```

3377 \newcommand*{\GlsXtrSetActualChar}[1]{%
3378 \gdef\@glsxtr@autoindex@at{#1}%
3379 \def\@glsxtr@autoindex@escat##1##2##3\@glsxtr@endescspch{%
3380     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escat}{##1}{##2}{##3}%
3381 }%
3382 }
3383 \@onlypreamble\GlsXtrSetActualChar
3384 \makeatother
3385 \GlsXtrSetActualChar{@}

```

```

3386 \makeatletter

autoindex@encap  Encap character for use with \index.
3387 \newcommand*{\@glxtr@autoindex@encap}{\}

XtrSetEncapChar  Set the encap character.
3388 \newcommand*{\GlsXtrSetEncapChar}[1]{%
3389   \gdef\@glxtr@autoindex@encap{#1}%
3390   \def\@glxtr@autoindex@escencap##1##2##3\@glxtr@endescspch{%
3391     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escencap}{##1}{##2}{##3}%
3392   }%
3393 }
3394 \GlsXtrSetEncapChar{||}
3395 \@onlypreamble\GlsXtrSetEncapChar

autoindex@level  Level character for use with \index.
3396 \newcommand*{\@glxtr@autoindex@level}{\}

XtrSetLevelChar  Set the encap character.
3397 \newcommand*{\GlsXtrSetLevelChar}[1]{%
3398   \gdef\@glxtr@autoindex@level{#1}%
3399   \def\@glxtr@autoindex@esclevel##1##2##3\@glxtr@endescspch{%
3400     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@esclevel}{##1}{##2}{##3}%
3401   }%
3402 }
3403 \GlsXtrSetLevelChar{!}
3404 \@onlypreamble\GlsXtrSetLevelChar

r@autoindex@esc  Escape character for use with \index.
3405 \newcommand*{\@glxtr@autoindex@esc}{"}

lsXtrSetEscChar  Set the escape character.
3406 \newcommand*{\GlsXtrSetEscChar}[1]{%
3407   \gdef\@glxtr@autoindex@esc{#1}%
3408   \def\@glxtr@autoindex@escquote##1##2##3\@glxtr@endescspch{%
3409     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escquote}{##1}{##2}{##3}%
3410   }%
3411 }
3412 \GlsXtrSetEscChar{"}
3413 \@onlypreamble\GlsXtrSetEscChar

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:
3414 \ifdef\actualchar
3415   {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
3416 {}

Quote character \quotechar:
3417 \ifdef\quotechar
3418   {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
3419 {}

```

Level character \levelchar:

```
3420 \ifdef\levelchar
3421 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
3422 {}
```

Encap character \encapchar:

```
3423 \ifdef\encapchar
3424 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
3425 {}
```

leto@endescspch

```
3426 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

toindex@esc@spch

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

```
3427 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
3428   \@glstmpb=\expandafter{\@glscmdkidx}%
3429   \toks@={#3}%
3430   \ifx\@nnil#3\relax
3431     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
3432   \else
3433     \ifx\@nnil#4\relax
3434       \edef\@glscmdkidx{\the\@glstmpb\the\toks@}%
3435       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
3436         #4#5\@glxtr@endescspch}%
3437     \else
3438       \edef\@glscmdkidx{\the\@glstmpb\the\toks@
3439         \@glxtr@autoindex@esc#1}%
3440       \def\@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
3441     \fi
3442   \fi
3443   \@glxtr@checkspch
3444 }
```

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

```
3445 \renewcommand*\Glossentrydesc[1]{%
3446   \glsoifexistsorwarn{#1}%
3447   {%
3448     \glsssetabbrvfmt{\glscategory{#1}}%
3449     \Glsaccessdesc{#1}%
3450   }%
3451 }
```

lossentrysymbol Redefine to set the abbreviation format and accessibility support.

```
3452 \renewcommand*\lossentrysymbol[1]{%
3453   \glsoifexistsorwarn{#1}%
```

```

3454  {%
3455    \glssetabbrvfmt{\glscategory{#1}}%
3456    \glsaccesssymbol{#1}%
3457  }%
3458 }

```

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

```

3459 \renewcommand*{\Glossentrysymbol}[1]{%
3460   \glsdoifexistsorwarn{#1}%
3461   {%
3462     \glssetabbrvfmt{\glscategory{#1}}%
3463     \Glsaccesssymbol{#1}%
3464   }%
3465 }

```

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.

```

3466 \newcommand*{\GlsXtrEnableInitialTagging}{%
3467   \@ifstar\s@glsxtr@enabletagging\@glsxtr@enabletagging
3468 }
3469 \@onlypreamble\GlsXtrEnableInitialTagging

```

`\r@enabletagging` Starred version undefines command.

```

3470 \newcommand*{\s@glsxtr@enabletagging}[2]{%
3471   \undef#2%
3472   \@glsxtr@enabletagging{#1}{#2}%
3473 }

```

`\r@enabletagging` Internal command.

```

3474 \newcommand*{\@glsxtr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
3475   \@for\@glsxtr@cat:=#1\do
3476   {%
3477     \ifdefempty\@glsxtr@cat
3478     {}%
3479     {\glssetcategoryattribute{\@glsxtr@cat}{tagging}{true}}%
3480   }%
3481   \newrobustcmd*#2[1]{##1}%
3482   \def\@glsxtr@taggingcs{#2}%
3483   \renewcommand*\@glsxtr@activate@initialtagging{%
3484     \let#2\@glsxtr@tag
3485   }%
3486   \ifundef\@gls@preglossaryhook
3487   {\GlossariesExtraWarning{Initial tagging requires at least
3488     glossaries.sty v4.19 to work correctly}}%

```

```

3489  {}%
3490 }

```

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

```

3491 \ifundef\mfu@checkword@do
3492 {
3493   \newcommand*{\mfu@checkword@do}[1]{%
3494     \ifdefstring{\mfu@checkword@arg}{#1}%
3495     {%
3496       \let\@mfu@domakefirstuc\@firstofone
3497       \listbreak
3498     }%
3499   }%
3500 }

```

\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.

```

3501 \ifundef\mfu@checkword
3502 {
3503   \newcommand{\@glxstr@do@titlecaps@warn}{%
3504     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
3505       support not available}%

```

One warning should suffice.

```

3506     \let\@glxstr@do@titlecaps@warn\relax
3507   }
3508 }
3509 {
3510   \renewcommand*{\mfu@checkword}[1]{%
3511     \def\mfu@checkword@arg{#1}%
3512     \let\@mfu@domakefirstuc\makefirstuc
3513     \forlistloop\mfu@checkword@do\@mfu@nocaplist
3514   }
3515 }
3516 }
3517 {}% no patch required

```

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

```

3518 \newcommand*{\@glxstr@do@titlecaps@warn}{}

```

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

```

3519 \newcommand*\@glxstr@activate@initialtagging{}

```

\@glxstr@tag Definition of tagging command when used in glossary.

```

3520 \newrobustcmd*{\@glxstr@tag}[1]{%
3521   \glscurrententrylabel}{tagging}{true}%

```



```

3522  {\glxstrtagfont{#1}}{#1}%
3523 }

```

`\glxstrtagfont` Used in the glossary.

```

3524 \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.

```

3525 \ifdef\@gls@preglossaryhook
3526 {
3527   \renewcommand*{\@gls@preglossaryhook}{%
3528     \@glxtr@activate@initialtagging
3529     \let\@glxtr@org@postdescription\glspostdescription
3530     \renewcommand*{\glspostdescription}{%
3531       \ifglstryexists{\glscurrententrylabel}%
3532       {%
3533         \glxtrpostdescription
3534         \@glxtr@org@postdescription
3535       }{}%
3536     }%

```

Enable the options used by `\@glxtrp`:

```

3537   \glossxtrsetpopts
3538 }%
3539 }
3540 {}

```

`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.

```

3541 \newcommand*{\glxtrpostdescription}{%
3542   \csuse{glxtrpostdesc\glscategory{\glscurrententrylabel}}%
3543 }

```

`postdescgeneral`

```

3544 \newcommand*{\glxtrpostdescgeneral}{}

```

`xtrpostdescterm`

```

3545 \newcommand*{\glxtrpostdescterm}{}

```

`postdescacronym`

```

3546 \newcommand*{\glxtrpostdescacronym}{}

```

`escabbreviation`

```

3547 \newcommand*{\glxtrpostdescabbreviation}{}

```

`glspostlinkhook` Redefine the post link hook used by commands like `\gls` to make it easier for categories or attributes to modify this action. Since this hook occurs outside the existence check of commands like `\gls`, this needs to be checked again here. Do nothing if the entry hasn't been defined.

```
3548 \renewcommand*{\glspostlinkhook}{%
3549   \ifglstryexists{\glslabel}{\glstrpostlinkhook}{}%
3550 }
```

`glstrpostlinkhook` The entry label should already be stored in `\glslabel` by `\@gls@link`.

```
3551 \newcommand*{\glstrpostlinkhook}{%
3552   \glstrdiscardperiod{\glslabel}%
3553   {\glstrpostlinkendsentence}%
3554   {\glstrpostlink}%
3555 }
```

`\glstrpostlink`

```
3556 \newcommand*{\glstrpostlink}{%
3557   \csuse{glstrpostlink\glscategory{\glslabel}}%
3558 }
```

`linkendsentence` Done by `\glstrpostlinkhook` if a full stop is discarded.

```
3559 \newcommand*{\glstrpostlinkendsentence}{%
3560   \ifcsdef{glstrpostlink\glscategory{\glslabel}}
3561   {%
3562     \csuse{glstrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
3563     .\spacefactor\sfcode'\. \relax
3564   }%
3565   {%
```

Assume the full stop was discarded because the entry ends with a period, so adjust the spacefactor.

```
3566     \spacefactor\sfcode'\. \relax
3567   }%
3568 }
```

`addDescOnFirstUse` Provide a command for appending the description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
3569 \newcommand*{\glstrpostlinkAddDescOnFirstUse}{%
3570   \glstrifwasfirstuse{\space(\glsaccessdesc{\glslabel})}{}%
3571 }
```

`symbolOnFirstUse` Provide a command for appending the symbol (if defined) in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
3572 \newcommand*{\glstrpostlinkAddSymbolOnFirstUse}{%
3573   \glstrifwasfirstuse
3574   {%
```

```

3575 \ifglshassymbol{\glslabel}{\space(\glssaccesssymbol{\glslabel})}{}%
3576 }%
3577 {}%
3578 }

```

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).

```

3579 \newcommand*{\glxtrdiscardperiod}[3]{%
3580 \glxtrifwasfirstuse
3581 {%
3582 \glusifattribute{#1}{retainfirstuseperiod}{true}%
3583 {#3}%
3584 {%
3585 \glusifattribute{#1}{discardperiod}{true}%
3586 {%
3587 \glusifplural
3588 {%
3589 \glusifattribute{#1}{pluraldiscardperiod}{true}%
3590 {\glxtrifperiod{#2}{#3}}%
3591 {#3}%
3592 }%
3593 {%
3594 \glxtrifperiod{#2}{#3}%
3595 }%
3596 }%
3597 {#3}%
3598 }%
3599 }%
3600 {%
3601 \glusifattribute{#1}{discardperiod}{true}%
3602 {%
3603 \glusifplural
3604 {%
3605 \glusifattribute{#1}{pluraldiscardperiod}{true}%
3606 {\glxtrifperiod{#2}{#3}}%
3607 {#3}%
3608 }%
3609 {%
3610 \glxtrifperiod{#2}{#3}%
3611 }%
3612 }%
3613 {#3}%
3614 }%
3615 }

```

\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

```
3616 \newcommand*{\glxstrifperiod}[1]{\new@ifnextchar.{\@firstoftwo{#1}}}
```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`\glxstr@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 ' ').

```
3617 \newcommand*{\glxstr@punclist}{.,:;?!}
```

`\punctuationmark` Add character to punctuation list.

```
3618 \newcommand*{\glxstraddpunctuationmark}[1]{\appto\glxstr@punclist{#1}}
```

`\punctuationmarks` Reset the punctuation list.

```
3619 \newcommand*{\glxstrsetpunctuationmarks}[1]{\def\glxstr@punclist{#1}}
```

```
\glxstrifpunc \glxstrifnextpunc{<true part>}{<false part>}
```

Test if this is followed by a punctuation mark. (Adapted from `\new@ifnextchar`.)

```
3620 \newcommand*{\glxstrifnextpunc}[2]{%
3621   \def\reserved@a{#1}%
3622   \def\reserved@b{#2}%
3623   \futurelet\@glspunc@token\glxstr@ifnextpunc
3624 }
```

`\glxstr@ifnextpunc`

```
3625 \newcommand*{\glxstr@ifnextpunc}{%
3626   \glxstr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
3627   \reserved@b
3628 }
```

`\glxstr@ifpunctoken` Test if the token given in the first argument is in the punctuation list.

```
3629 \newcommand*{\glxstr@ifpunctoken}[1]{%
3630   \expandafter\@glxstr@ifpunctoken\expandafter#1\glxstr@punclist\@nnil
3631 }
```

`\glxstr@ifpunctoken`

```
3632 \def\@glxstr@ifpunctoken#1#2{%
3633   \let\reserved@d=#2%
3634   \ifx\reserved@d\@nnil
3635     \let\glxstr@next\@glxstr@notfoundinlist
3636   \else
3637     \ifx#1\reserved@d
3638       \let\glxstr@next\@glxstr@foundinlist
3639     \else
3640       \let\glxstr@next\@glxstr@ifpunctoken
3641     \fi
3642   \fi
```

```

3643 \glxtr@next#1%
3644 }

```

xtr@foundinlist

```

3645 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}

```

@notfoundinlist

```

3646 \def\@glxtr@notfoundinlist#1{\@secondoftwo}

```

glxtrdopostpunc

```
\glxtrdopostpunc{<code>}
```

If this is followed by a punctuation character, do *<code>* after the character otherwise do *<code>* before whatever comes next.

```

3647 \newcommand{\glxtrdopostpunc}[1]{%
3648 \glxtrifnextpunc{\@glxtr@swaptwo{#1}}{#1}%
3649 }

```

@glxtr@swaptwo

```

3650 \newcommand{\@glxtr@swaptwo}[2]{#2#1}

```

1.6 Abbreviations

The “acronym” code from glossaries is misnamed as it’s more often used for other forms of abbreviations. This code corrects this inconsistency, but rather than just having synonyms, provide commands for abbreviations that have a similar, but not identical, underlying mechanism to acronyms.

If there’s a style for the given category, apply it.

```

3651 \define@key{glxtrabbrv}{category}{%
3652 \edef\glscategorylabel{#1}%
3653 \ifcsdef{@glxabbrv@current@#1}%
3654 {%

```

Warning should already have been issued.

```

3655 \let\@glxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
3656 \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
3657 \glxtr@applyabbrvstyle{\csname @glxabbrv@current@#1\endcsname}%
3658 \let\GlsXtrWarnDeprecatedAbbrStyle\@glxtr@orgwarndep
3659 }%
3660 {}%
3661 }

```

Save the short plural form. This may be needed before the entry is defined.

```

3662 \define@key{glxtrabbrv}{shortplural}{%
3663 \def\@glx@shortpl{#1}%
3664 }

```

Similarly for the long plural form.

```
3665 \define@key{glxtrabbrv}{longplural}{%
3666   \def\@gls@longpl{#1}%
3667 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```
3668 \newtoks\glsshortpltok
```

\glslongpltok

```
3669 \newtoks\glslongpltok
```

`glxtr@insertdots` Provided in case user wants to automatically insert dots between each letter of the abbreviation. This should be applied before defining the abbreviation to optimise the document build. (Otherwise, it would have to be done each time the short form is required, which is an unnecessary waste of time.) For this to work the short form must be expanded when passed to `\newabbreviation`. Note that explicitly using the short or shortplural keys will override this.

```
3670 \newcommand*{\@glxtr@insertdots}[2]{%
3671   \def#1{%
3672     \@glxtr@insert@dots#1#2\@nnil
3673 }
```

`glxtr@insert@dots`

```
3674 \newcommand*{\@glxtr@insert@dots}[2]{%
3675   \ifx\@nnil#2\relax
3676   \let\@glxtr@insert@dots@next\@gobble
3677   \else
3678     \ifx\relax#2\relax
3679     \else
3680       \appto#1{#2.}%
3681     \fi
3682     \let\@glxtr@insert@dots@next\@glxtr@insert@dots
3683   \fi
3684   \@glxtr@insert@dots@next#1%
3685 }
```

`newabbreviation` Define a new generic abbreviation.

```
3686 \newcommand*{\newabbreviation}[4][]{%
3687   \glskeylisttok{#1}%
3688   \glslabeltok{#2}%
3689   \glsshorttok{#3}%
3690   \glslongtok{#4}%
```

Get the category.

```
3691   \def\glscategorylabel{abbreviation}%
3692   \glxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
3693   \setkeys*{glxtrabbrv}{shortplural,longplural}{#1}%
```

Set the default long plural

```
3694 \def\@gls@longpl{#4\glspluralsuffix}%
```

Has the insertdots attribute been set?

```
3695 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
3696 {%
3697   \@glsxtr@insertdots\@gls@short{#3}%
3698   \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
3699   \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
3700   {%
3701     \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
3702       '\abbrvpluralsuffix}%
3703   }%
3704   {%
3705     \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
3706     {%
3707       \let\@gls@shortpl\@gls@short
3708     }%
3709     {%
3710       \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
3711         \abbrvpluralsuffix}%
3712     }%
3713   }%
3714 }%
3715 {%
```

insertdots not true.

```
3716 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
3717 {%
3718   \def\@gls@shortpl{#3'\abbrvpluralsuffix}%
3719 }%
3720 {%
3721   \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
3722   {%
3723     \def\@gls@shortpl{#3}%
3724   }%
3725   {%
3726     \def\@gls@shortpl{#3\abbrvpluralsuffix}%
3727   }%
3728 }%
3729 }%
```

Hook for further customisation if required:

```
3730 \glsxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%
```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary.

```
3731 \setkeys*{glsxtrabbrv}[category]{#1}%
```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```
3732 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
```

3733 \expandafter\glslongpltok\expandafter{\@gls@longpl}%

Do any extra setup provided by hook:

3734 \newabbreviationhook

Define this entry:

```

3735 \protected@edef\@do@newglossaryentry{%
3736   \noexpand\newglossaryentry{\the\glslabeltok}%
3737   {%
3738     type=\glsxtrabbrvtype,%
3739     category=abbreviation,%
3740     short={\the\glsshorttok},%
3741     shortplural={\the\glsshortpltok},%
3742     long={\the\glslongtok},%
3743     longplural={\the\glslongpltok},%
3744     name={\the\glsshorttok},%
3745     \CustomAbbreviationFields,%
3746     \the\glskeylisttok
3747   }%
3748 }%
3749 \@do@newglossaryentry
3750 \GlsXtrPostNewAbbreviation
3751 }
```

evpresetkeyhook Hook for extra stuff in \newabbreviation

3752 \newcommand*{\glsxtrnewabbrevpresetkeyhook}[3]{}%

NewAbbreviation Hook used by abbreviation styles.

3753 \newcommand*{\GlsXtrPostNewAbbreviation}{}%

bbreviationhook Hook for use with \newabbreviation.

3754 \newcommand*{\newabbreviationhook}{}%

reviationFields

3755 \newcommand*{\CustomAbbreviationFields}{}%

lsxtrfullformat Full format without case change.

```

3756 \newcommand*{\glsxtrfullformat}[2]{%
3757   \glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
3758   (\protect\glsfirstabbrvfont{\glsaccessshort{#1}})%
3759 }
```

lsxtrfullformat Full format with case change.

```

3760 \newcommand*{\Glsxtrfullformat}[2]{%
3761   \glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
3762   (\protect\glsfirstabbrvfont{\Glsaccessshort{#1}})%
3763 }
```


`\glxtrfullplformat` Plural full format without case change.

```
3764 \newcommand*{\glxtrfullplformat}[2]{%
3765   \glsfirstlongfont{\glssaccesslongpl{#1}}#2\glxtrfullsep{#1}%
3766   (\protect\glsfirstabbrvfont{\glssaccessshortpl{#1}})%
3767 }
```

`\glxtrfullplformat` Plural full format with case change.

```
3768 \newcommand*{\Glsxtrfullplformat}[2]{%
3769   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glxtrfullsep{#1}%
3770   (\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}})%
3771 }
```

`\glxtrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
3772 \newcommand*{\glxtrfullsep}[1]{\space}
```

In-line formats in case first use isn't compatible with `\gl Sentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glxtrinlinefullformat` Full format without case change.

```
3773 \newcommand*{\glxtrinlinefullformat}{\glxtrfullformat}
```

`\glxtrinlinefullformat` Full format with case change.

```
3774 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}
```

`\glxtrfullplformat` Plural full format without case change.

```
3775 \newcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}
```

`\glxtrinlinefullplformat` Plural full format with case change.

```
3776 \newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}
```

Redefine `\gl Sentryfull` etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the `\glxtrfull` set of commands instead.

`\gl Sentryfull`

```
3777 \renewcommand*{\gl Sentryfull}[1]{\glxtrinlinefullformat{#1}{}}
```

`\Glsentryfull`

```
3778 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\gl Sentryfullpl`

```
3779 \renewcommand*{\gl Sentryfullpl}[1]{\glxtrinlinefullplformat{#1}{}}
```

`\Glsentryfullpl`

```
3780 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\gl sfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
3781 \newcommand*{\gl sfirstabbrvfont}[1]{\gl sfirstabbrvdefaultfont{#1}}
```

```

\brvdefaultfont  Font changing command used for the abbreviation on first use or in the full format.
3782 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}

\glsabbrvfont  Font changing command used for the abbreviation on subsequent use.
3783 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}

\brvdefaultfont
3784 \newcommand*{\glsabbrvdefaultfont}[1]{#1}

\glslongfont  Font changing command used for the long form in commands like \glsxtrlong.
3785 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}

\glslongdefaultfont  Default font changing command used for the long form in commands like \glsxtrlong.
3786 \newcommand*{\glslongdefaultfont}[1]{#1}

\glsfirstlongfont  Font changing command used for the long form on first use or in the full format.
3787 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}

\glslongdefaultfont
3788 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}

\brvpluralsuffix  Default plural suffix.
3789 \newcommand*{\abbrvpluralsuffix}{\glspluralsuffix}

\glsxtrfull  Full form (no case-change).
3790 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\@ns@glsxtrfull}
3791 \newcommand*{\ns@glsxtrfull}[2][\%]
3792   \new@ifnextchar[{\@glsxtr@full{#1}{#2}}]{\%}
3793   {\@glsxtr@full{#1}{#2}}[\%]
3794 }

\@glsxtr@full  Low-level macro:
3795 \def\@glsxtr@full#1#2[#3]{\%
3796   \glsdoifexists{#2}\%
3797   {\%
3798     \glssetabbrvfmt{\glscategory{#2}}\%
3799     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3800     \let\glsifplural\@secondoftwo
3801     \let\glsupcase\@firstofthree
3802     \let\glsinsert\@empty
3803     \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
different. Provide a hook to make it easier to reconfigure.

3804   \glsxtrsetupfulldefs
3805   \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}\%

```

```

3806 }%
3807 \glspostlinkhook
3808 }

```

trsetupfulldefs

```

3809 \newcommand*{\glxtrsetupfulldefs}{%
3810   \let\glxtrifwasfirstuse\@firstoftwo
3811 }

```

\Glsxtrfull Full form (first letter uppercase).

```

3812 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\@ns@Glsxtrfull}
3813 \newcommand*\ns@Glsxtrfull[2][{}]{%
3814   \new@ifnextchar[\@Glsxtr@full{#1}{#2}}%
3815   {\@Glsxtr@full{#1}{#2}[]}%
3816 }

```

\@Glsxtr@full Low-level macro:

```

3817 \def\@Glsxtr@full#1#2[#3]{%
3818   \glsdoifexists{#2}%
3819   {%
3820     \glsssetabbrvfmt{\glscategory{#2}}%
3821     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3822     \let\glsifplural\@secondoftwo
3823     \let\glscapscase\@secondofthree
3824     \let\glsinsert\@empty
3825     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
3826     \glxtrsetupfulldefs
3827     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3828   }%
3829   \glspostlinkhook
3830 }

```

\GLSxtrfull Full form (all uppercase).

```

3831 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\@ns@GLSxtrfull}
3832 \newcommand*\ns@GLSxtrfull[2][{}]{%
3833   \new@ifnextchar[\@GLSxtr@full{#1}{#2}}%
3834   {\@GLSxtr@full{#1}{#2}[]}%
3835 }

```

\@GLSxtr@full Low-level macro:

```

3836 \def\@GLSxtr@full#1#2[#3]{%
3837   \glsdoifexists{#2}%
3838   {%
3839     \glsssetabbrvfmt{\glscategory{#2}}%
3840     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3841     \let\glsifplural\@secondoftwo
3842     \let\glscapscase\@thirdofthree
3843     \let\glsinsert\@empty
3844     \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%

```

```

3845 \glstrsetupfulldefs
3846 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3847 }%
3848 \glspostlinkhook
3849 }

```

`\glstrfullpl` Plural full form (no case-change).

```

3850 \newrobustcmd*{\glstrfullpl}{\@gls@hyp@opt\ns@glstrfullpl}
3851 \newcommand*\ns@glstrfullpl[2][{}]{%
3852 \new@ifnextchar[{\@glstr@fullpl{#1}{#2}}%
3853 {\@glstr@fullpl{#1}{#2}[]}%
3854 }

```

`\@glstr@fullpl` Low-level macro:

```

3855 \def\@glstr@fullpl#1#2[#3]{%
3856 \glsdoifexists{#2}%
3857 {%
3858 \glsetabbrvfmt{\glscategory{#2}}%
3859 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3860 \let\glsifplural\@firstoftwo
3861 \let\glscapscase\@firstofthree
3862 \let\glsinsert\@empty
3863 \def\glscustomtext{\glstrinlinefullplformat{#2}{#3}}%
3864 \glstrsetupfulldefs
3865 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3866 }%
3867 \glspostlinkhook
3868 }

```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```

3869 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\ns@Glsxtrfullpl}
3870 \newcommand*\ns@Glsxtrfullpl[2][{}]{%
3871 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
3872 {\@Glsxtr@fullpl{#1}{#2}[]}%
3873 }

```

`\@Glsxtr@fullpl` Low-level macro:

```

3874 \def\@Glsxtr@fullpl#1#2[#3]{%
3875 \glsdoifexists{#2}%
3876 {%
3877 \glsetabbrvfmt{\glscategory{#2}}%
3878 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3879 \let\glsifplural\@firstoftwo
3880 \let\glscapscase\@secondofthree
3881 \let\glsinsert\@empty
3882 \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
3883 \glstrsetupfulldefs
3884 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3885 }%

```

```

3886 \glspostlinkhook
3887 }

```

`\GLSxtrfullpl` Plural full form (all upper case).

```

3888 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\ns@GLSxtrfullpl}
3889 \newcommand*\ns@GLSxtrfullpl[2] [] {%
3890   \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}]{%
3891     {\@GLSxtr@fullpl{#1}{#2} []}%
3892 }

```

`\@GLSxtr@fullpl` Low-level macro:

```

3893 \def\@GLSxtr@fullpl#1#2[#3] {%
3894   \glsdoifexists{#2}%
3895   {%
3896     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3897     \let\glsifplural\@firstoftwo
3898     \let\glscapscase\@thirdofthree
3899     \let\glsinsert\@empty
3900     \def\glscustomtext{%
3901       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
3902     \glsxtrsetupfulldefs
3903     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3904   }%
3905   \glspostlinkhook
3906 }

```

The short and long forms work in a similar way to acronyms.

`\glsxtrshort`

```

3907 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3908 \newcommand*{\ns@glsxtrshort}[2] [] {%
3909   \new@ifnextchar[{\@glsxtrshort{#1}{#2}}{\@glsxtrshort{#1}{#2} []}%
3910 }

```

Read in the final optional argument:

```

3911 \def\@glsxtrshort#1#2[#3] {%
3912   \glsdoifexists{#2}%
3913   {%

```

Need to make sure `\glsabbrvfont` is set correctly.

```

3914   \glssetabbrvfmt{\glscategory{#2}}%
3915   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3916   \let\glsxtrifwasfirstuse\@secondoftwo
3917   \let\glsifplural\@secondoftwo
3918   \let\glscapscase\@firstofthree
3919   \let\glsinsert\@empty
3920   \def\glscustomtext{%
3921     \glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%

```

```

3922     \ifglxtrinsertinside\else#3\fi
3923   }%
3924   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3925   }%
3926   \glspostlinkhook
3927 }

```

\Glsxtrshort

```

3928 \newrobustcmd*{\Glsxtrshort}{\@gls@hyp@opt\ns@Glsxtrshort}

  Define the un-starred form. Need to determine if there is a final optional argument
3929 \newcommand*{\ns@Glsxtrshort}[2] [] {%
3930   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} []}%
3931 }

  Read in the final optional argument:
3932 \def\@Glsxtrshort#1#2[#3] {%
3933   \glsdoifexists{#2}%
3934   {%
3935     \glsetabbrvfmt{\glscategory{#2}}%
3936     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3937     \let\glxtrifwasfirstuse\@secondoftwo
3938     \let\glsifplural\@secondoftwo
3939     \let\glscapscase\@secondofthree
3940     \let\glsinsert\@empty
3941     \def\glscustomtext{%
3942       \glsabbrvfont{\Glsaccessshort{#2}\ifglxtrinsertinside#3\fi}%
3943       \ifglxtrinsertinside\else#3\fi
3944     }%
3945     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3946   }%
3947   \glspostlinkhook
3948 }

```

\GLSxtrshort

```

3949 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\ns@GLSxtrshort}

  Define the un-starred form. Need to determine if there is a final optional argument
3950 \newcommand*{\ns@GLSxtrshort}[2] [] {%
3951   \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} []}%
3952 }

  Read in the final optional argument:
3953 \def\@GLSxtrshort#1#2[#3] {%
3954   \glsdoifexists{#2}%
3955   {%
3956     \glsetabbrvfmt{\glscategory{#2}}%
3957     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3958     \let\glxtrifwasfirstuse\@secondoftwo
3959     \let\glsifplural\@secondoftwo
3960     \let\glscapscase\@thirdofthree

```

```

3961 \let\glsinsert\@empty
3962 \def\glscustomtext{%
3963   \mfirstucMakeUppercase
3964   {\glsabbrvfont{\glsaccessshort{#2}\ifglstrinsertinside#3\fi}%
3965   \ifglstrinsertinside\else#3\fi
3966   }%
3967   }%
3968   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3969   }%
3970 \glspostlinkhook
3971 }

```

\glstrlong

```

3972 \newrobustcmd*{\glstrlong}{\@gls@hyp@opt\@ns@glstrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
3973 \newcommand*{\ns@glstrlong}[2][{}]{%
3974   \new@ifnextchar[{\@glstrlong{#1}{#2}}{\@glstrlong{#1}{#2}[]}%
3975 }

    Read in the final optional argument:
3976 \def\@glstrlong#1#2[#3]{%
3977   \glsdoifexists{#2}%
3978   {%
3979     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3980     \let\glstrifwasfirstuse\@secondoftwo
3981     \let\glstrifplural\@secondoftwo
3982     \let\glscapscase\@firstofthree
3983     \let\glsinsert\@empty
3984     \def\glscustomtext{%
3985       \glsfont{\glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
3986       \ifglstrinsertinside\else#3\fi
3987     }%
3988     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3989     }%
3990     \glspostlinkhook
3991 }

```

\Glsxtrlong

```

3992 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
3993 \newcommand*{\ns@Glsxtrlong}[2][{}]{%
3994   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2}[]}%
3995 }

    Read in the final optional argument:
3996 \def\@Glsxtrlong#1#2[#3]{%
3997   \glsdoifexists{#2}%
3998   {%
3999     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```

```

4000 \let\glxstrifwasfirstuse\@secondoftwo
4001 \let\glxifplural\@secondoftwo
4002 \let\glscapscase\@secondofthree
4003 \let\glxinsert\@empty
4004 \def\glscustomtext{%
4005   \glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
4006   \ifglxtrinsertinside\else#3\fi
4007 }%
4008 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4009 }%
4010 \glspostlinkhook
4011 }

```

\GLSxtrlong

```

4012 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
4013 \newcommand*{\ns@GLSxtrlong}[2] [] {%
4014   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
4015 }
    Read in the final optional argument:
4016 \def\@GLSxtrlong#1#2[#3] {%
4017   \glsdoidexists{#2}%
4018   {%
4019     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
4020     \let\glxstrifwasfirstuse\@secondoftwo
4021     \let\glxifplural\@secondoftwo
4022     \let\glscapscase\@thirdofthree
4023     \let\glxinsert\@empty
4024     \def\glscustomtext{%
4025       \mfirstucMakeUppercase
4026       {\glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
4027       \ifglxtrinsertinside\else#3\fi
4028     }%
4029   }%
4030   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4031 }%
4032 \glspostlinkhook
4033 }

```

Plural short forms:

\glxtrshortpl

```

4034 \newrobustcmd*{\glxtrshortpl}{\@gls@hyp@opt\ns@glxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
4035 \newcommand*{\ns@glxtrshortpl}[2] [] {%
4036   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2} []}%
4037 }

```


Read in the final optional argument:

```

4038 \def\@glxstrshortpl#1#2[#3]{%
4039   \glstoifexists{#2}%
4040   {%
4041     \glsetabbrvfmt{\glscategory{#2}}%
4042     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4043     \let\glxtrifwasfirstuse\@secondoftwo
4044     \let\gl@sifplural\@firstoftwo
4045     \let\glscapscase\@firstofthree
4046     \let\glinsert\@empty
4047     \def\glscustomtext{%
4048       \glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
4049       \ifglxtrininsertinside\else#3\fi
4050     }%
4051     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4052   }%
4053   \glspostlinkhook
4054 }

```

\Glsxtrshortpl

```

4055 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\ns@Glsxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4056 \newcommand*{\ns@Glsxtrshortpl}[2][ ]{%
4057   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2}[ ]}%
4058 }

```

Read in the final optional argument:

```

4059 \def\@Glsxtrshortpl#1#2[#3]{%
4060   \glstoifexists{#2}%
4061   {%
4062     \glsetabbrvfmt{\glscategory{#2}}%
4063     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
4064     \let\glxtrifwasfirstuse\@secondoftwo
4065     \let\gl@sifplural\@firstoftwo
4066     \let\glscapscase\@secondofthree
4067     \let\glinsert\@empty
4068     \def\glscustomtext{%
4069       \glabbrvfont{\Glsaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
4070       \ifglxtrininsertinside\else#3\fi
4071     }%
4072     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4073   }%
4074   \glspostlinkhook
4075 }

```

\GLSxtrshortpl

```

4076 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\ns@GLSxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4077 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
4078   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
4079 }

```

Read in the final optional argument:

```

4080 \def\@GLSxtrshortpl#1#2[#3]{%
4081   \glsdoifexists{#2}%
4082   {%
4083     \glssetabbrvfmt{\glscategory{#2}}%
4084     \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
4085     \let\glsxtrifwasfirstuse\@secondoftwo
4086     \let\glsifplural\@firstoftwo
4087     \let\glscapscase\@thirdofthree
4088     \let\glsinsert\@empty
4089     \def\glscustomtext{%
4090       \mfirstucMakeUppercase
4091       {\glsabbrvfont{\glsaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
4092       \ifglxtrininsertinside\else#3\fi
4093     }%
4094   }%
4095   \@gl@s@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4096 }%
4097 \glspostlinkhook
4098 }

```

Plural long forms:

`\glsxtrlongpl`

```

4099 \newrobustcmd*{\glsxtrlongpl}{\@gl@s@hyp@opt\@ns@glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

4100 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
4101   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
4102 }

```

Read in the final optional argument:

```

4103 \def\@glsxtrlongpl#1#2[#3]{%
4104   \glsdoifexists{#2}%
4105   {%
4106     \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
4107     \let\glsxtrifwasfirstuse\@secondoftwo
4108     \let\glsifplural\@firstoftwo
4109     \let\glscapscase\@firstofthree
4110     \let\glsinsert\@empty
4111     \def\glscustomtext{%
4112       \glsfont{\glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
4113       \ifglxtrininsertinside\else#3\fi
4114     }%
4115     \@gl@s@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4116   }%
4117   \glspostlinkhook

```

4118 }

\Glsxtrlongpl

4119 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}

Define the un-starred form. Need to determine if there is a final optional argument

4120 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%

4121 \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%

4122 }

Read in the final optional argument:

4123 \def\@Glsxtrlongpl#1#2[#3]{%

4124 \glsdoifexists{#2}%

4125 {%

4126 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

4127 \let\glxtrifwasfirstuse\@secondoftwo

4128 \let\glsifplural\@firstoftwo

4129 \let\glscapscase\@secondofthree

4130 \let\glsinsert\@empty

4131 \def\glscustomtext{%

4132 \glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%

4133 \ifglxtrininsertinside\else#3\fi

4134 }%

4135 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%

4136 }%

4137 \glspostlinkhook

4138 }

\GLSxtrlongpl

4139 \newrobustcmd*{\GLSxtrlongpl}{\@gls@hyp@opt\@ns@GLSxtrlongpl}

Define the un-starred form. Need to determine if there is a final optional argument

4140 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%

4141 \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%

4142 }

Read in the final optional argument:

4143 \def\@GLSxtrlongpl#1#2[#3]{%

4144 \glsdoifexists{#2}%

4145 {%

4146 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

4147 \let\glxtrifwasfirstuse\@secondoftwo

4148 \let\glsifplural\@firstoftwo

4149 \let\glscapscase\@thirdofthree

4150 \let\glsinsert\@empty

4151 \def\glscustomtext{%

4152 \mfirstucMakeUppercase

4153 {\glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%

4154 \ifglxtrininsertinside\else#3\fi

4155 }%

4156 }%

```

4157 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
4158 }%
4159 \glspostlinkhook
4160 }

```

`\glssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

4161 \newcommand*{\glssetabbrvfmt}[1]{%
4162 \ifcsdef{@glsabbrv@current@#1}%
4163 {\glsxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
4164 {\glsxtr@applyabbrvfmt{@glsabbrv@current@abbreviation}}%
4165 }

```

`\glstrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

4166 \newcommand*{\glstrgenabbrvfmt}{%
4167 \ifdefempty\glscustomtext
4168 {%
4169 \ifglsused\glslabel
4170 {%

```

Subsequent use:

```

4171 \glsifplural
4172 {%

```

Subsequent plural form:

```

4173 \glscapscase
4174 {%

```

Subsequent plural form, don't adjust case:

```

4175 \glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert
4176 }%
4177 {%

```

Subsequent plural form, make first letter upper case:

```

4178 \glsabbrvfont{\Glsaccessshortpl{\glslabel}}\glsinsert
4179 }%
4180 {%

```

Subsequent plural form, all caps:

```

4181 \mfirstucMakeUppercase
4182 {\glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert}%
4183 }%
4184 }%
4185 {%

```

Subsequent singular form

```

4186 \glscapscase
4187 {%

```

Subsequent singular form, don't adjust case:

```

4188 \glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert
4189 }%
4190 {%

```

Subsequent singular form, make first letter upper case:

```
4191      \glsabbrvfont{\Glsaccesssshort{\glslabel}}\glsinsert
4192      }%
4193      {%
```

Subsequent singular form, all caps:

```
4194      \mfirstucMakeUppercase
4195      {\glsabbrvfont{\glsaccesssshort{\glslabel}}\glsinsert}%
4196      }%
4197      }%
4198      }%
4199      {%
```

First use:

```
4200      \glsifplural
4201      {%
```

First use plural form:

```
4202      \glscapscase
4203      {%
```

First use plural form, don't adjust case:

```
4204      \glstrfullplformat{\glslabel}{\glsinsert}%
4205      }%
4206      {%
```

First use plural form, make first letter upper case:

```
4207      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
4208      }%
4209      {%
```

First use plural form, all caps:

```
4210      \mfirstucMakeUppercase
4211      {\glstrfullplformat{\glslabel}{\glsinsert}}%
4212      }%
4213      }%
4214      {%
```

First use singular form

```
4215      \glscapscase
4216      {%
```

First use singular form, don't adjust case:

```
4217      \glstrfullformat{\glslabel}{\glsinsert}%
4218      }%
4219      {%
```

First use singular form, make first letter upper case:

```
4220      \Glsxtrfullformat{\glslabel}{\glsinsert}%
4221      }%
4222      {%
```

First use singular form, all caps:

```

4223         \mfirstucMakeUppercase
4224         {\glstrfullformat{\glslabel}{\glsinsert}}%
4225     }%
4226 }%
4227 }%
4228 }%
4229 {%

```

User supplied text.

```

4230     \glscustomtext
4231 }%
4232 }

```

1.6.1 Abbreviation Styles Setup

abbreviationstyle

```

4233 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
4234     \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
4235     {%
4236         \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
4237     }%
4238     {%

```

Have abbreviations already been defined for this category?

```

4239     \ifcsstring{@glsabbrv@current@#1}{#2}%
4240     {%

```

Style already set.

```

4241     }%
4242     {%
4243         \def\@glsxtr@dostylewarn{}%
4244         \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
4245         {%
4246             \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
4247                 style has been switched \MessageBreak
4248                 for category ‘#1’, \MessageBreak
4249                 but there have already been entries \MessageBreak
4250                 defined for this category. Unwanted \MessageBreak
4251                 side-effects may result}}%
4252             \@endfortrue
4253         }%
4254         \@glsxtr@dostylewarn

```

Set up the style for the given category.

```

4255         \csdef{@glsabbrv@current@#1}{#2}%
4256         \glsxtr@applyabbrvstyle{#2}%
4257     }%
4258 }%
4259 }

```

`\applyabbrvstyle` Apply the abbreviation style without existence check.

```
4260 \newcommand*{\glxtr@applyabbrvstyle}[1]{%
4261   \csuse{@glsabbrv@dispstyle@setup@#1}%
4262   \csuse{@glsabbrv@dispstyle@fmts@#1}%
4263 }
```

`\r@applyabbrvfmt` Only apply the style formats.

```
4264 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
4265   \csuse{@glsabbrv@dispstyle@fmts@#1}%
4266 }
```

`\abbreviationstyle` This is different from `\newacronymstyle`. The first argument is the label, the second argument sets the information required when defining the new abbreviation and the third argument sets the commands used to display the full format.

```
4267 \newcommand*{\newabbreviationstyle}[3]{%
4268   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
4269     {%
4270       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
4271         defined}{}%
4272     }%
4273     {%
4274       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
4275       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
4276       #2}%
4277       \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
4278       \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
4279       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
4280       \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
4281       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
4282       #3}%
4283     }%
4284 }
```

`\renewabbreviationstyle`

```
4285 \newcommand*{\renewabbreviationstyle}[3]{%
4286   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
4287     {%
4288       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
4289     }%
4290     {%
4291       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
4292       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
4293       #2}%
4294       \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```

4295 \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
4296 \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
4297 \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
4298 \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
4299 #3}%
4300 }%
4301 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

4302 \newcommand*{\letabbreviationstyle}[2]{%
4303 \csletcs{@glssabrv@dispstyle@setup@#1}{@glssabrv@dispstyle@setup@#2}%
4304 \csletcs{@glssabrv@dispstyle@fmts@#1}{@glssabrv@dispstyle@fmts@#2}%
4305 }

```

deprecated@abbrstyle `\@glxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

4306 \newcommand*{\@glxtr@deprecated@abbrstyle}[2]{%
4307 \csdef{@glssabrv@dispstyle@setup@#1}{%
4308 \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
4309 \csuse{@glssabrv@dispstyle@setup@#2}%
4310 }%
4311 \csletcs{@glssabrv@dispstyle@fmts@#1}{@glssabrv@dispstyle@fmts@#2}%
4312 }

```

deprecatedAbbrStyle Generate warning for deprecated style use.

```

4313 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
4314 \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
4315 use ‘#2’ instead}%
4316 }

```

UseAbbrStyleSetup

```

4317 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
4318 \ifcsundef{@glssabrv@dispstyle@setup@#1}%
4319 {%
4320 \PackageError{glossaries-extra}%
4321 {Unknown abbreviation style definitions ‘#1’}{}%
4322 }%
4323 {%
4324 \csname @glssabrv@dispstyle@setup@#1\endcsname
4325 }%
4326 }

```


seAbbrStyleFmts

```
4327 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
4328   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
4329   {%
4330     \PackageError{glossaries-extra}%
4331       {Unknown abbreviation style formats ‘#1’}{}%
4332   }%
4333   {%
4334     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
4335   }%
4336 }
```

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.

```
4337 \newif\ifglsxtrinsertinside
4338 \glsxtrinsertinsidefalse
```

long-short

```
4339 \newabbreviationstyle{long-short}%
4340 {%
4341   \renewcommand*{\CustomAbbreviationFields}{%
4342     name={\protect\glsabbrvfont{\the\glsshorttok}},
4343     sort={\the\glsshorttok},
4344     first={\protect\glsfirstlongfont{\the\glslongtok}%
4345       \protect\glsxtrfullsep{\the\glslabeltok}%
4346       (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
4347     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
4348       \protect\glsxtrfullsep{\the\glslabeltok}%
4349       (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%
4350     plural={\protect\glsabrvfont{\the\glsshortpltok}},%
4351     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
4352 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4353   \glshasattribute{\the\glslabeltok}{regular}%
4354   {%
4355     \glssetattribute{\the\glslabeltok}{regular}{false}%
4356   }%
```

```

4357 {}%
4358 }%
4359 }%
4360 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4361 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4362 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4363 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4364 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4365 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

4366 \renewcommand*{\glsxtrfullformat}[2]{%
4367   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
4368   \ifglsxtrinsertinside\else##2\fi
4369   \glsxtrfullsep{##1}%
4370   (\glsfirstabbrvfont{\glsaccessshort{##1}})%
4371 }%
4372 \renewcommand*{\glsxtrfullplformat}[2]{%
4373   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
4374   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4375   (\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4376 }%
4377 \renewcommand*{\Glsxtrfullformat}[2]{%
4378   \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
4379   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4380   (\glsfirstabbrvfont{\glsaccessshort{##1}})%
4381 }%
4382 \renewcommand*{\Glsxtrfullplformat}[2]{%
4383   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
4384   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4385   (\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4386 }%
4387 }

```

Set this as the default style for general abbreviations:

```

4388 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

4389 \newcommand*{\glsxtrlongshortdescsort}{\the\glslongtok\space(\the\glsshorttok)}

```

long-short-desc User supplies description. The long form is included in the name.

```

4390 \newabbreviationstyle{long-short-desc}%
4391 {%
4392   \renewcommand*{\CustomAbbreviationFields}{%
4393     name={\protect\glsxtrfullformat{\the\glslabeltok}{}},
4394     sort={\glsxtrlongshortdescsort},%
4395     first={\protect\glsfirstlongfont{\the\glslongtok}}%
4396     \protect\glsxtrfullsep{\the\glslabeltok}%

```

```

4397      (\protect\glsfirstabbrvfont{\the\glsshorttok}}),%
4398      firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%
4399      \protect\glstrfullsep{\the\glslabeltok}%
4400      (\protect\glsfirstabbrvfont{\the\glsshortpltok}}),%

```

The text key should only have the short form.

```

4401      text={\protect\glsabbrvfont{\the\glsshorttok}},%
4402      plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
4403  }%

```

Unset the regular attribute if it has been set.

```

4404  \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4405    \glshasattribute{\the\glslabeltok}{regular}%
4406    {%
4407      \glsssetAttribute{\the\glslabeltok}{regular}{false}%
4408    }%
4409    {}%
4410  }%
4411 }%
4412 {%
4413   \GlsXtrUseAbbrStyleFmts{long-short}%
4414 }

```

short-long Short form followed by long form in parenthesis on first use.

```

4415 \newabbreviationstyle{short-long}%
4416 {%
4417   \renewcommand*{\CustomAbbreviationFields}{%
4418     name={\protect\glsabbrvfont{\the\glsshorttok}},
4419     sort={\the\glsshorttok},
4420     description={\the\glslongtok},%
4421     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
4422     \protect\glstrfullsep{\the\glslabeltok}%
4423     (\protect\glsfirstlongfont{\the\glslongtok}}),%
4424     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
4425     \protect\glstrfullsep{\the\glslabeltok}%
4426     (\protect\glsfirstlongfont{\the\glslongpltok}}),%
4427     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

4428  \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4429    \glshasattribute{\the\glslabeltok}{regular}%
4430    {%
4431      \glsssetAttribute{\the\glslabeltok}{regular}{false}%
4432    }%
4433    {}%
4434  }%
4435 }%
4436 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4437 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4438 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4439 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4440 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4441 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

The first use full form and the inline full form are the same for this style.

4442 \renewcommand*{\glsxtrfullformat}[2]{%
4443   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
4444   \ifglsxtrinsertinside\else##2\fi
4445   \glsxtrfullsep{##1}%
4446   (\glsfirstlongfont{\glsaccesslong{##1}})%
4447 }%
4448 \renewcommand*{\glsxtrfullplformat}[2]{%
4449   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
4450   \ifglsxtrinsertinside\else##2\fi
4451   \glsxtrfullsep{##1}%
4452   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4453 }%
4454 \renewcommand*{\Glsxtrfullformat}[2]{%
4455   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
4456   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4457   (\glsfirstlongfont{\glsaccesslong{##1}})%
4458 }%
4459 \renewcommand*{\Glsxtrfullplformat}[2]{%
4460   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
4461   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4462   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4463 }%
4464 }

```

short-long-desc User supplies description. The long form is included in the name.

```

4465 \newabbreviationstyle{short-long-desc}%
4466 {%
4467   \renewcommand*{\CustomAbbreviationFields}{%
4468     name={\protect\glsxtrfullformat{\the\glslabeltok}{}},
4469     sort={\the\glsshorttok},%
4470     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
4471       \protect\glsxtrfullsep{\the\glslabeltok}%
4472       (\protect\glsfirstlongfont{\the\glslongtok})},%
4473     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
4474       \protect\glsxtrfullsep{\the\glslabeltok}%
4475       (\protect\glsfirstlongfont{\the\glslongpltok})},%
4476     text={\protect\glsabbrvfont{\the\glsshorttok}},%
4477     plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
4478   }%

```

Unset the regular attribute if it has been set.

```

4479 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4480   \glshasattribute{\the\glslabeltok}{regular}%
4481   {%
4482     \glissetattribute{\the\glslabeltok}{regular}{false}%
4483   }%
4484   {}%
4485 }%
4486 }%
4487 {%
4488   \GlsXtrUseAbbrStyleFmts{short-long}%
4489 }

```

ongfootnotefont Only used by the “footnote” styles.

```

4490 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%

```

ongfootnotefont Only used by the “footnote” styles.

```

4491 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%

```

xtrabbrvfootnote

`\glsxtrabbrvfootnote{<label>}{<long>}`

Command used by footnote abbreviation styles. The default definition ignores the first argument. The second argument *<long>* includes the font changing command and may be the singular or plural form, depending on the command that was used (for example, `\gls` or `\glspl`).

```

4492 \newcommand*{\glsxtrabbrvfootnote}[2]{\footnote{#2}}

```

footnote Short form followed by long form in footnote on first use.

```

4493 \newabbreviationstyle{footnote}%
4494 {%
4495   \renewcommand*{\CustomAbbreviationFields}{%
4496     name={\protect\glsabbrvfont{\the\glsshorttok}},
4497     sort={\the\glsshorttok},
4498     description={\the\glslongtok},%
4499     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
4500     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
4501     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
4502     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
4503     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
4504     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
4505     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.

```

4506 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

4507 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
4508 \glsattribute{\the\glslabeltok}{regular}%
4509 {%
4510 \glssetattribute{\the\glslabeltok}{regular}{false}%
4511 }%
4512 {}%
4513 }%
4514 }%
4515 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4516 \renewcommand*\abbrvpluralsuffix{\glspluralsuffix}%
4517 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
4518 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
4519 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
4520 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

4521 \renewcommand*\glsxtrfullformat[2]{%
4522 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4523 \ifglsxtrininsertinside\else##2\fi
4524 \protect\glsxtrabbrvfootnote{##1}%
4525 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
4526 }%
4527 \renewcommand*\glsxtrfullplformat[2]{%
4528 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4529 \ifglsxtrininsertinside\else##2\fi
4530 \protect\glsxtrabbrvfootnote{##1}%
4531 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
4532 }%
4533 \renewcommand*\Glsxtrfullformat[2]{%
4534 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4535 \ifglsxtrininsertinside\else##2\fi
4536 \protect\glsxtrabbrvfootnote{##1}%
4537 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
4538 }%
4539 \renewcommand*\Glsxtrfullplformat[2]{%
4540 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4541 \ifglsxtrininsertinside\else##2\fi
4542 \protect\glsxtrabbrvfootnote{##1}%
4543 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
4544 }%

```

The first use full form and the inline full form use the short (long) style.

```

4545 \renewcommand*\glsxtrinlinefullformat[2]{%
4546 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
4547 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4548 (\glsfirstlongfootnotefont{\glsaccesslong{##1}})%
4549 }%
4550 \renewcommand*\glsxtrinlinefullplformat[2]{%
4551 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%

```

```

4552 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4553 (\glsfirstlongfootnotefont{\glssaccesslongpl{##1}})%
4554 }%
4555 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4556 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4557 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4558 (\glsfirstlongfootnotefont{\glssaccesslong{##1}})%
4559 }%
4560 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4561 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4562 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4563 (\glsfirstlongfootnotefont{\glssaccesslongpl{##1}})%
4564 }%
4565 }

```

short-footnote

```

4566 \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.

```

4567 \newabbreviationstyle{postfootnote}%
4568 {%
4569 \renewcommand*{\CustomAbbreviationFields}{%
4570 name={\protect\glssabbrvfont{\the\glssshorttok}},
4571 sort={\the\glssshorttok},
4572 description={\the\glslongtok},%
4573 first={\protect\glsfirstabbrvfont{\the\glssshorttok}},%
4574 firstplural={\protect\glsfirstabbrvfont{\the\glssshortpltok}},%
4575 plural={\protect\glssabbrvfont{\the\glssshortpltok}}}%

```

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.

```

4576 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4577 \csdef{glxtrpostlink\glscategorylabel}{%
4578 \glxtrifwasfirstuse
4579 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

4580 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
4581 {\glsfirstlongfootnotefont{\glssentrylong{\glslabel}}}}%
4582 }%
4583 {}%
4584 }%
4585 \glshasattribute{\the\glslabeltok}{regular}%
4586 {%
4587 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
4588 }%

```

```

4589   {}%
4590 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

4591 \renewcommand*{\glxtrsetupfulldefs}{%
4592   \let\glxtrifwasfirstuse\@secondoftwo
4593 }%
4594 }%
4595 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4596 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4597 \renewcommand*{\glabbrvfont}[1]{\glabbrvdefaultfont{##1}}%
4598 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvdefaultfont{##1}}%
4599 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongfootnotefont{##1}}%
4600 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

4601 \renewcommand*{\glxtrfullformat}[2]{%
4602   \glfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
4603   \ifglxtrininsertinside\else##2\fi
4604 }%
4605 \renewcommand*{\glxtrfullplformat}[2]{%
4606   \glfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
4607   \ifglxtrininsertinside\else##2\fi
4608 }%
4609 \renewcommand*{\Glsxtrfullformat}[2]{%
4610   \glfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
4611   \ifglxtrininsertinside\else##2\fi
4612 }%
4613 \renewcommand*{\Glsxtrfullplformat}[2]{%
4614   \glfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
4615   \ifglxtrininsertinside\else##2\fi
4616 }%

```

The first use full form and the inline full form use the short (long) style.

```

4617 \renewcommand*{\glxtrinlinefullformat}[2]{%
4618   \glfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
4619   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
4620   (\glfirstlongfootnotefont{\glsaccesslong{##1}})%
4621 }%
4622 \renewcommand*{\glxtrinlinefullplformat}[2]{%
4623   \glfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
4624   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
4625   (\glfirstlongfootnotefont{\glsaccesslongpl{##1}})%
4626 }%
4627 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4628   \glfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
4629   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
4630   (\glfirstlongfootnotefont{\Glsaccesslong{##1}})%

```



```

4631 }%
4632 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4633   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4634   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4635   (\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}})%
4636 }%
4637 }

```

rt-postfootnote

```

4638 \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.

```

4639 \newabbreviationstyle{short}%
4640 {%
4641   \renewcommand*{\CustomAbbreviationFields}{%
4642     name={\protect\glsabbrvfont{\the\glsshorttok}},
4643     sort={\the\glsshorttok},
4644     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
4645     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
4646     text={\protect\glsabbrvfont{\the\glsshorttok}},
4647     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
4648     description={\the\glslongtok}}%
4649   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4650     \glssetattribute{\the\glslabeltok}{regular}{true}}%
4651 }%
4652 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4653 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4654 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4655 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4656 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4657 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

4658 \renewcommand*{\glxtrinlinefullformat}[2]{%
4659   \protect\glsfirstabbrvfont{\Glsaccessshort{##1}}%
4660   \ifglxtrinsertinside##2\fi}%
4661   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4662   (\glsfirstlongfont{\Glsaccesslong{##1}})%
4663 }%
4664 \renewcommand*{\glxtrinlinefullplformat}[2]{%
4665   \protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}}%
4666   \ifglxtrinsertinside##2\fi}%
4667   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4668   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
4669 }%

```

```

4670 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4671   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
4672   \ifglsxtrininsertinside##2\fi}%
4673   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4674   (\glsfirstlongfont{\Glsaccesslong{##1}})%
4675 }%
4676 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4677   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
4678   \ifglsxtrininsertinside##2\fi}%
4679   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4680   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
4681 }%

```

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.

```

4682 \renewcommand*{\glsxtrfullformat}[2]{%
4683   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
4684   \ifglsxtrininsertinside\else##2\fi
4685 }%
4686 \renewcommand*{\glsxtrfullplformat}[2]{%
4687   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
4688   \ifglsxtrininsertinside\else##2\fi
4689 }%
4690 \renewcommand*{\Glsxtrfullformat}[2]{%
4691   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
4692   \ifglsxtrininsertinside\else##2\fi
4693 }%
4694 \renewcommand*{\Glsxtrfullplformat}[2]{%
4695   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
4696   \ifglsxtrininsertinside\else##2\fi
4697 }%
4698 }

```

Set this as the default style for acronyms:

```

4699 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

4700 \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.

```

4701 \newabbreviationstyle{short-desc}%
4702 {%
4703   \renewcommand*{\CustomAbbreviationFields}{%
4704     name={\protect\glsxtrinlinefullformat{\the\glslabeltok}{}},
4705     sort={\the\glsshorttok},
4706     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
4707     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
4708     text={\protect\glsabbrvfont{\the\glsshorttok}},

```

```

4709 plural={\protect\glsabbrvfont{\the\glsshortpltok}},
4710 description={\the\glslongtok}}%
4711 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4712   \glssetattribute{\the\glslabeltok}{regular}{true}}%
4713 }%
4714 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4715 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4716 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4717 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4718 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4719 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

4720 \renewcommand*{\glsxtrinlinefullformat}[2]{%
4721   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
4722   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4723   (\glsfirstlongfont{\glsaccesslong{##1}})%
4724 }%
4725 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
4726   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
4727   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4728   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4729 }%
4730 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4731   \glsfirstabbrvfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
4732   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4733   (\glsfirstlongfont{\Glsaccesslong{##1}})%
4734 }%
4735 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4736   \glsfirstabbrvfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
4737   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4738   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
4739 }%

```

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.

```

4740 \renewcommand*{\glsxtrfullformat}[2]{%
4741   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
4742   \ifglsxtrininsertinside\else##2\fi
4743 }%
4744 \renewcommand*{\glsxtrfullplformat}[2]{%
4745   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
4746   \ifglsxtrininsertinside\else##2\fi
4747 }%
4748 \renewcommand*{\Glsxtrfullformat}[2]{%
4749   \glsfirstabbrvfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
4750   \ifglsxtrininsertinside\else##2\fi
4751 }%
4752 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

4753 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4754 \ifglxtrinsertinside\else##2\fi
4755 }%
4756 }

```

ort-nolong-desc

```
4757 \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.

```

4758 \newabbreviationstyle{long-desc}%
4759 {%
4760 \renewcommand*{\CustomAbbreviationFields}{%
4761 name={\protect\protect\glsfirstlongfont{\the\glslongtok}},
4762 sort={\the\glslongtok},
4763 first={\protect\glsfirstlongfont{\the\glslongtok}},
4764 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
4765 text={\the\glslongtok},
4766 plural={\the\glslongpltok}}%
4767 }%
4768 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4769 \glssetattribute{\the\glslabeltok}{regular}{true}}%
4770 }%
4771 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4772 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4773 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4774 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4775 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
4776 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the long format followed by the short form in parentheses.

```

4777 \renewcommand*{\glxtrinlinefullformat}[2]{%
4778 \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
4779 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4780 (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
4781 }%
4782 \renewcommand*{\glxtrinlinefullplformat}[2]{%
4783 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
4784 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4785 (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4786 }%
4787 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4788 \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
4789 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4790 (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
4791 }%
4792 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%

```

```

4793 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
4794 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4795 (\protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}})%
4796 }%

```

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.

```

4797 \renewcommand*{\glxtrfullformat}[2]{%
4798 \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
4799 \ifglxtrinsertinside\else##2\fi
4800 }%
4801 \renewcommand*{\glxtrfullplformat}[2]{%
4802 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
4803 \ifglxtrinsertinside\else##2\fi
4804 }%
4805 \renewcommand*{\Glsxtrfullformat}[2]{%
4806 \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
4807 \ifglxtrinsertinside\else##2\fi
4808 }%
4809 \renewcommand*{\Glsxtrfullplformat}[2]{%
4810 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
4811 \ifglxtrinsertinside\else##2\fi
4812 }%
4813 }

```

ng-noshort-desc Provide a synonym that matches similar styles.

```

4814 \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.

```

4815 \newabbreviationstyle{long}%
4816 {%
4817 \renewcommand*{\CustomAbbreviationFields}{%
4818 name={\protect\glsabbrvfont{\the\glsshorttok}},
4819 sort={\the\glsshorttok},
4820 first={\protect\glsfirstlongfont{\the\glslongtok}},
4821 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
4822 text={\the\glslongtok},
4823 plural={\the\glslongpltok},%
4824 description={\the\glslongtok}%
4825 }%
4826 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4827 \glssetattribute{\the\glslabeltok}{regular}{true}}%
4828 }%
4829 {%
4830 \GlsXtrUseAbbrStyleFmts{long-desc}%
4831 }

```

long-noshort Provide a synonym that matches similar styles.

```
4832 \letabbreviationstyle{long-noshort}{long}
```

1.6.3 Predefined Styles (Small Capitals)

These styles use:

\glxtrscfont

```
4833 \newcommand*{\glxtrscfont}[1]{\textsc{#1}}
```

sxtrfirstscfont

```
4834 \newcommand*{\glxsxtrfirstscfont}[1]{\glxtrscfont{#1}}
```

and for the default short form suffix:

\glxtrscsuffix

```
4835 \newcommand*{\glxtrscsuffix}{\glstextup{\glspluralsuffix}}
```

long-short-sc

```
4836 \newabbreviationstyle{long-short-sc}%  
4837 {%  
4838   \GlsXtrUseAbbrStyleSetup{long-short}%  
4839 }%  
4840 {%
```

Mostly as long-short style:

```
4841   \GlsXtrUseAbbrStyleFmts{long-short}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4842   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
4843   \renewcommand*{\glsabbrvfont}[1]{\glxtrscfont{##1}}%  
4844   \renewcommand*{\glsfirstabbrvfont}[1]{\glxsxtrfirstscfont{##1}}%  
4845 }
```

g-short-sc-desc

```
4846 \newabbreviationstyle{long-short-sc-desc}%  
4847 {%  
4848   \GlsXtrUseAbbrStyleSetup{long-short-desc}%  
4849 }%  
4850 {%
```

Mostly as long-short-desc style:

```
4851   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4852   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
4853   \renewcommand*{\glsabbrvfont}[1]{\glxtrscfont{##1}}%  
4854   \renewcommand*{\glsfirstabbrvfont}[1]{\glxsxtrfirstscfont{##1}}%  
4855 }
```

Now the short (long) version

```
4856 \newabbreviationstyle{short-sc-long}%  
4857 {%  
4858   \GlsXtrUseAbbrStyleSetup{short-long}%  
4859 }%  
4860 {%
```

Mostly as short-long style:

```
4861   \GlsXtrUseAbbrStyleFmts{short-long}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4862   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
4863   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%  
4864   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%  
4865 }
```

As before but user provides description

```
4866 \newabbreviationstyle{short-sc-long-desc}%  
4867 {%  
4868   \GlsXtrUseAbbrStyleSetup{short-long-desc}%  
4869 }%  
4870 {%
```

Mostly as short-long-desc style:

```
4871   \GlsXtrUseAbbrStyleFmts{short-long-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4872   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
4873   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%  
4874   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%  
4875 }
```

short-sc

```
4876 \newabbreviationstyle{short-sc}%  
4877 {%  
4878   \GlsXtrUseAbbrStyleSetup{short-nolong}%  
4879 }%  
4880 {%
```

Mostly as short style:

```
4881   \GlsXtrUseAbbrStyleFmts{short-nolong}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4882   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
4883   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%  
4884   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%  
4885 }
```

short-sc-nolong

```
4886 \letabbreviationstyle{short-sc-nolong}{short-sc}
```

```

short-sc-desc
4887 \newabbreviationstyle{short-sc-desc}%
4888 {%
4889   \GlsXtrUseAbbrStyleSetup{short-desc}%
4890 }%
4891 {%
    Mostly as short style:
4892   \GlsXtrUseAbbrStyleFmts{short-desc}%
    Use smallcaps and adjust the plural suffix to revert to upright.
4893   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4894   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%
4895   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%
4896 }

-sc-nolong-desc
4897 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

long-noshort-sc  The smallcaps font will only be used if the short form is explicitly invoked through commands
                  like \glsshort.
4898 \newabbreviationstyle{long-noshort-sc}%
4899 {%
4900   \GlsXtrUseAbbrStyleSetup{long-noshort}%
4901 }%
4902 {%
    Mostly as long style:
4903   \GlsXtrUseAbbrStyleFmts{long-noshort}%
    Use smallcaps and adjust the plural suffix to revert to upright.
4904   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4905   \renewcommand*\glsabbrvfont[1]{\glxtrscfont{##1}}%
4906   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstscfont{##1}}%
4907 }

long-sc  Backward compatibility:
4908 \@glxtr@deprecated@abbrstyle{long-sc}{long-noshort-sc}

noshort-sc-desc  The smallcaps font will only be used if the short form is explicitly invoked through commands
                  like \glsshort.
4909 \newabbreviationstyle{long-noshort-sc-desc}%
4910 {%
4911   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
4912 }%
4913 {%
    Mostly as long style:
4914   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%

```


Use smallcaps and adjust the plural suffix to revert to upright.

```
4915 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4916 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
4917 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
4918 }
```

long-desc-sc Backward compatibility:

```
4919 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}
```

ort-sc-footnote

```
4920 \newabbreviationstyle{short-sc-footnote}%
4921 {%
4922   \GlsXtrUseAbbrStyleSetup{short-footnote}%
4923 }%
4924 {%
```

Mostly as long style:

```
4925 \GlsXtrUseAbbrStyleFmts{short-footnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4926 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4927 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
4928 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
4929 }
```

footnote-sc Backward compatibility:

```
4930 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}
```

sc-postfootnote

```
4931 \newabbreviationstyle{short-sc-postfootnote}%
4932 {%
4933   \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
4934 }%
4935 {%
```

Mostly as long style:

```
4936 \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4937 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4938 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
4939 \renewcommand*{\glssfirstabrvfont[1]{\glxtrfirstscfont{##1}}}%
4940 }
```

postfootnote-sc Backward compatibility:

```
4941 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}
```

1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the `relese` package, which must be loaded by the user. These styles all use:

`\glxtrsmfont`

```
4942 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}
```

`sxtrfirstsmfont`

```
4943 \newcommand*{\glxtrfirstsmfont}[1]{\glxtrsmfont{#1}}
```

and for the default short form suffix:

`\glxtrsmsuffix`

```
4944 \newcommand*{\glxtrsmsuffix}{\glspluralsuffix}
```

`long-short-sm`

```
4945 \newabbreviationstyle{long-short-sm}%  
4946 {%  
4947   \GlsXtrUseAbbrStyleSetup{long-short}%  
4948 }%  
4949 {%
```

Mostly as long-short style:

```
4950   \GlsXtrUseAbbrStyleFmts{long-short}%  
4951   \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%  
4952   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%  
4953   \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%  
4954 }
```

`g-short-sm-desc`

```
4955 \newabbreviationstyle{long-short-sm-desc}%  
4956 {%  
4957   \GlsXtrUseAbbrStyleSetup{long-short-desc}%  
4958 }%  
4959 {%
```

Mostly as long-short-desc style:

```
4960   \GlsXtrUseAbbrStyleFmts{long-short-desc}%  
4961   \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%  
4962   \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%  
4963   \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%  
4964 }
```

`short-sm-long` Now the short (long) version

```
4965 \newabbreviationstyle{short-sm-long}%  
4966 {%  
4967   \GlsXtrUseAbbrStyleSetup{short-long}%  
4968 }%  
4969 {%
```

Mostly as short-long style:

```
4970 \GlsXtrUseAbbrStyleFmts{short-long}%
4971 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
4972 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
4973 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
4974 }
```

rt-sm-long-desc As before but user provides description

```
4975 \newabbreviationstyle{short-sm-long-desc}%
4976 {%
4977   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
4978 }%
4979 {%
```

Mostly as short-long-desc style:

```
4980 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
4981 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
4982 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
4983 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
4984 }
```

short-sm

```
4985 \newabbreviationstyle{short-sm}%
4986 {%
4987   \GlsXtrUseAbbrStyleSetup{short-nolong}%
4988 }%
4989 {%
```

Mostly as short style:

```
4990 \GlsXtrUseAbbrStyleFmts{short-nolong}%
4991 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
4992 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
4993 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
4994 }
```

short-sm-nolong

```
4995 \letabbreviationstyle{short-sm-nolong}{short-sm}
```

short-sm-desc

```
4996 \newabbreviationstyle{short-sm-desc}%
4997 {%
4998   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
4999 }%
5000 {%
```

Mostly as short style:

```
5001 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
5002 \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
5003 \renewcommand*\glsfirstabbrvfont[1]{\glxtrfirstsmfont{##1}}%
5004 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
5005 }
```

```

-sm-nolong-desc
    5006 \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 \glssshort.
    5007 \newabbreviationstyle{long-noshort-sm}%
    5008 {%
    5009   \GlsXtrUseAbbrStyleSetup{long-noshort}%
    5010 }%
    5011 {%
        Mostly as long style:
    5012   \GlsXtrUseAbbrStyleFmts{long-noshort}%
    5013   \renewcommand*{\glsabbrvfont[1]{\glstrsmfont{##1}}}%
    5014   \renewcommand*{\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}}%
    5015   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrsmsuffix}%
    5016 }

long-sm  Backward compatibility:
    5017 \@glstr@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 \glssshort.
    5018 \newabbreviationstyle{long-noshort-sm-desc}%
    5019 {%
    5020   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
    5021 }%
    5022 {%
        Mostly as long style:
    5023   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
    5024   \renewcommand*{\glsabbrvfont[1]{\glstrsmfont{##1}}}%
    5025   \renewcommand*{\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}}%
    5026   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrsmsuffix}%
    5027 }

long-desc-sm  Backward compatibility:
    5028 \@glstr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

ort-sm-footnote
    5029 \newabbreviationstyle{short-sm-footnote}%
    5030 {%
    5031   \GlsXtrUseAbbrStyleSetup{short-footnote}%
    5032 }%
    5033 {%
        Mostly as long style:
    5034   \GlsXtrUseAbbrStyleFmts{short-footnote}%
    5035   \renewcommand*{\glsabbrvfont[1]{\glstrsmfont{##1}}}%

```

```

5036 \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}%
5037 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
5038 }

```

footnote-sm Backward compatibility:

```

5039 \@glstr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

sm-postfootnote

```

5040 \newabbreviationstyle{short-sm-postfootnote}%
5041 {%
5042   \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
5043 }%
5044 {%

```

Mostly as long style:

```

5045   \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
5046   \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
5047   \renewcommand*\glsfirstabbrvfont[1]{\glstrfirstsmfont{##1}}%
5048   \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
5049 }

```

postfootnote-sm Backward compatibility:

```

5050 \@glstr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```

5051 \newcommand*\glsabbrvemfont[1]{\emph{#1}}%

```

`irstabbrvemfont`

```

5052 \newcommand*\glsfirstabbrvemfont[1]{\glsabbrvemfont{#1}}%

```

`firstlongemfont` Only used by the “long-em” styles.

```

5053 \newcommand*\glsfirstlongemfont[1]{\glslongemfont{#1}}%

```

`\glslongemfont` Only used by the “long-em” styles.

```

5054 \newcommand*\glslongemfont[1]{\emph{#1}}%

```

`long-short-em`

```

5055 \newabbreviationstyle{long-short-em}%
5056 {%
5057   \GlsXtrUseAbbrStyleSetup{long-short}%
5058 }%
5059 {%

```

Mostly as long-short style:

```
5060 \GlsXtrUseAbbrStyleFmts{long-short}%
5061 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5062 }
```

g-short-em-desc

```
5063 \newabbreviationstyle{long-short-em-desc}%
5064 {%
5065 \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5066 }%
5067 {%
```

Mostly as long-short-desc style:

```
5068 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
5069 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5070 }
```

ong-em-short-em

```
5071 \newabbreviationstyle{long-em-short-em}%
5072 {%
  \glslongemfont is used in the description since \glsdesc doesn't set the style.
5073 \renewcommand*{\CustomAbbreviationFields}{%
5074   name={\protect\glsabbrvfont{\the\glsshorttok}},
5075   sort={\the\glsshorttok},
5076   first={\protect\glsfirstlongfont{\the\glslongtok}%
5077     \protect\glsxtrfullsep{\the\glslabeltok}%
5078     (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
5079   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
5080     \protect\glsxtrfullsep{\the\glslabeltok}%
5081     (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%
5082   plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
5083   description={\protect\glslongemfont{\the\glslongtok}}}%
  }
```

Unset the regular attribute if it has been set.

```
5084 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5085   \glshasattribute{\the\glslabeltok}{regular}%
5086   {%
5087     \glssetattribute{\the\glslabeltok}{regular}{false}%
5088   }%
5089   {}%
5090 }%
5091 }%
5092 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5093 \GlsXtrUseAbbrStyleFmts{long-short}%
5094 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5095 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5096 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
```

```

5097 \renewcommand*\glslongfont}[1]{\glslongemfont{##1}}%
5098 }

```

m-short-em-desc

```

5099 \newabbreviationstyle{long-em-short-em-desc}%
5100 {%
5101 \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5102 }%
5103 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5104 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
5105 \renewcommand*\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5106 \renewcommand*\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5107 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5108 \renewcommand*\glslongfont}[1]{\glslongemfont{##1}}%
5109 }

```

short-em-long Now the short (long) version

```

5110 \newabbreviationstyle{short-em-long}%
5111 {%
5112 \GlsXtrUseAbbrStyleSetup{short-long}%
5113 }%
5114 {%

```

Mostly as short-long style:

```

5115 \GlsXtrUseAbbrStyleFmts{short-long}%
5116 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5117 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5118 }

```

rt-em-long-desc As before but user provides description

```

5119 \newabbreviationstyle{short-em-long-desc}%
5120 {%
5121 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5122 }%
5123 {%

```

Mostly as short-long-desc style:

```

5124 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
5125 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5126 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5127 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5128 \renewcommand*\glslongfont}[1]{\glslongemfont{##1}}%
5129 }

```

hort-em-long-em

```

5130 \newabbreviationstyle{short-em-long-em}%
5131 {%

```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```
5132 \renewcommand*{\CustomAbbreviationFields}{%
5133   name={\protect\glsabbrvfont{\the\glsshorttok}},
5134   sort={\the\glsshorttok},
5135   description={\protect\glslongemfont{\the\glslongtok}},%
5136   first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
5137   \protect\glstrfullsep{\the\glslabeltok}%
5138   (\protect\glsfirstlongfont{\the\glslongtok}}),%
5139   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
5140   \protect\glstrfullsep{\the\glslabeltok}%
5141   (\protect\glsfirstlongfont{\the\glslongpltok}}),%
5142   plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```
5143 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5144   \glshasattribute{\the\glslabeltok}{regular}%
5145   {%
5146     \glissetattribute{\the\glslabeltok}{regular}{false}%
5147   }%
5148   {}%
5149 }%
5150}%
5151{%

```

In case the user wants to mix and match font styles, these are redefined here.

```
5152 \GlsXtrUseAbbrStyleFmts{short-long}%
5153 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5154 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5155 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5156 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5157}

```

em-long-em-desc

```
5158 \newabbreviationstyle{short-em-long-em-desc}%
5159 {%
5160   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5161 }%
5162 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
5163 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
5164 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
5165 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5166 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5167 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
5168}

```

short-em

```
5169 \newabbreviationstyle{short-em}%
5170 {%

```



```

5171 \GlsXtrUseAbbrStyleSetup{short-nolong}%
5172 }%
5173 {%
    Mostly as short style:
5174 \GlsXtrUseAbbrStyleFmts{short-nolong}%
5175 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5176 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5177 }

```

short-em-nolong

```

5178 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

5179 \newabbreviationstyle{short-em-desc}%
5180 {%
5181 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
5182 }%
5183 {%
    Mostly as short style:
5184 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
5185 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5186 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5187 }

```

-em-nolong-desc

```

5188 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

5189 \newabbreviationstyle{long-noshort-em}%
5190 {%
5191 \GlsXtrUseAbbrStyleSetup{long-noshort}%
5192 }%
5193 {%
    Mostly as long-noshort style:
5194 \GlsXtrUseAbbrStyleFmts{long-noshort}%
5195 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5196 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5197 }

```

long-em Backward compatibility:

```

5198 \@glsxtr@deprecated@abbrstyle{long-em}{long-noshort-em}

```

g-em-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

5199 \newabbreviationstyle{long-em-noshort-em}%
5200 {%
5201 \renewcommand*\CustomAbbreviationFields{%

```

```

5202     name={\protect\glsabbrvfont{\the\glsshorttok}},
5203     sort={\the\glsshorttok},
5204     first={\protect\glsfirstlongfont{\the\glslongtok}},
5205     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
5206     text={\the\glslongtok},
5207     plural={\the\glslongpltok},%
5208     description={\protect\glslongemfont{\the\glslongtok}}%
5209 }%
5210 \renewcommand*\GlsXtrPostNewAbbreviation{%
5211   \glssetattribute{\the\glslabeltok}{regular}{true}}%
5212 }%
5213 {%

```

Mostly as long-noshort style:

```

5214   \GlsXtrUseAbbrStyleFmts{long-noshort}%
5215   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5216   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5217   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
5218   \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%
5219 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

5220 \newabbreviationstyle{long-noshort-em-desc}%
5221 {%
5222   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5223 }%
5224 {%

```

Mostly as long style:

```

5225   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
5226   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5227   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
5228 }

```

long-desc-em Backward compatibility:

```

5229 \@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.

```

5230 \newabbreviationstyle{long-em-noshort-em-desc}%
5231 {%
5232   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
5233 }%
5234 {%

```

Mostly as long style:

```

5235   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
5236   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5237   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%

```

```

5238 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
5239 \renewcommand*\glslongfont}[1]{\glslongemfont{##1}}%
5240 }

```

ort-em-footnote

```

5241 \newabbreviationstyle{short-em-footnote}%
5242 {%
5243   \GlsXtrUseAbbrStyleSetup{short-footnote}%
5244 }%
5245 {%
  Mostly as long style:
5246   \GlsXtrUseAbbrStyleFmts{short-footnote}%
5247   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5248   \renewcommand*\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5249 }

```

footnote-em Backward compatibility:

```

5250 \@glsxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

em-postfootnote

```

5251 \newabbreviationstyle{short-em-postfootnote}%
5252 {%
5253   \GlsXtrUseAbbrStyleSetup{short-postfootnote}%
5254 }%
5255 {%
  Mostly as long style:
5256   \GlsXtrUseAbbrStyleFmts{short-postfootnote}%
5257   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
5258   \renewcommand*\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
5259 }

```

postfootnote-em Backward compatibility:

```

5260 \@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.

```

5261 \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.

```

5262 \ifdef\glscurrentfieldvalue
5263 {

```

```

5264 \newcommand*{\glxtruserparen}[2]{%
5265   \glxtrfullsep{#2}%
5266   (#1\ifglshasfield{\glxtruserfield}{#2}{, \glscurrentfieldvalue}{})%
5267 }
5268 }
5269 {
5270 \newcommand*{\glxtruserparen}[2]{%
5271   \glxtrfullsep{#2}%
5272   (#1\ifglshasfield{\glxtruserfield}{#2}{, \@glo@thisvalue}{})%
5273 }
5274 }

```

Font used for short form:

lsabbrvuserfont

```
5275 \newcommand*{\glabbrvuserfont}[1]{#1}
```

Font used for short form on first use:

stabbrvuserfont

```
5276 \newcommand*{\glfirstabbrvuserfont}[1]{\glabbrvuserfont{#1}}
```

Font used for long form:

glslonguserfont

```
5277 \newcommand*{\glslonguserfont}[1]{#1}
```

Font used for long form on first use:

rstlonguserfont

```
5278 \newcommand*{\glfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
5279 \newcommand*{\glxtrusersuffix}{\glspuralsuffix}
```

long-short-user

```
5280 \newabbreviationstyle{long-short-user}%
5281 {%

```

\glslonguserfont is used in the description since \glsdsc doesn't set the style.

```

5282 \renewcommand*{\CustomAbbreviationFields}{%
5283   name={\protect\glabbrvfont{\the\glsshorttok}},
5284   sort={\the\glsshorttok},
5285   first={\protect\glfirstlongfont{\the\glslongtok}%
5286     \protect\glxtruserparen{\protect\glfirstabbrvfont{\the\glsshorttok}}{\the\glslabeltok}},
5287   firstplural={\protect\glfirstlongfont{\the\glslongtok}%
5288     \protect\glxtruserparen{\protect\glfirstabbrvfont{\the\glsshorttok}}{\the\glslabeltok}},
5289   plural={\protect\glabbrvfont{\the\glsshorttok}},%
5290   description={\protect\glslonguserfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

5291 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5292   \glshasattribute{\the\glslabeltok}{regular}%
5293   {%
5294     \glissetattribute{\the\glslabeltok}{regular}{false}%
5295   }%
5296   {}%
5297 }%
5298 }%
5299 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5300 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
5301 \renewcommand*{\glssabrvfont}[1]{\glssabrvuserfont{##1}}%
5302 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvuserfont{##1}}%
5303 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlonguserfont{##1}}%
5304 \renewcommand*{\glsslongfont}[1]{\glsslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5305 \renewcommand*{\glsxtrfullformat}[2]{%
5306   \glssfirstlongfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5307   \ifglsxtrininsertinside\else##2\fi
5308   \glsxtruserparen{\glssfirstabbrvfont{\glssaccessshort{##1}}}{##1}%
5309 }%
5310 \renewcommand*{\glsxtrfullplformat}[2]{%
5311   \glssfirstlongfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
5312   \ifglsxtrininsertinside\else##2\fi
5313   \glsxtruserparen{\glssfirstabbrvfont{\glssaccessshortpl{##1}}}{##1}%
5314 }%
5315 \renewcommand*{\Glsxtrfullformat}[2]{%
5316   \glssfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
5317   \ifglsxtrininsertinside\else##2\fi
5318   \glsxtruserparen{\glssfirstabbrvfont{\glssaccessshort{##1}}}{##1}%
5319 }%
5320 \renewcommand*{\Glsxtrfullplformat}[2]{%
5321   \glssfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
5322   \ifglsxtrininsertinside\else##2\fi
5323   \glsxtruserparen{\glssfirstabbrvfont{\glssaccessshortpl{##1}}}{##1}%
5324 }%
5325 }

```

short-user-desc

```

5326 \newabbreviationstyle{long-short-user-desc}%
5327 {%
5328   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
5329 }%
5330 {%
5331   \GlsXtrUseAbbrStyleFmts{long-short-user}%
5332 }

```

short-long-user

```
5333 \newabbreviationstyle{short-long-user}%  
5334 {%
```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```
5335 \renewcommand*{\CustomAbbreviationFields}{%  
5336   name={\protect\glsabbrvfont{\the\glsshorttok}},  
5337   sort={\the\glsshorttok},  
5338   description={\protect\glslonguserfont{\the\glslongtok}},%  
5339   first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%  
5340   \protect\glsxtruserparen{\protect\glslonguserfont{\the\glslongtok}}{\the\glslabeltok}},%  
5341   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%  
5342   \protect\glsxtruserparen{\protect\glslonguserfont{\the\glslongpltok}}{\the\glslabeltok}},%  
5343   plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
```

Unset the regular attribute if it has been set.

```
5344 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
5345   \glsasattribute{\the\glslabeltok}{regular}%  
5346   {%  
5347     \glssetattribute{\the\glslabeltok}{regular}{false}%  
5348   }}%  
5349 {}%  
5350 }%  
5351 }%  
5352 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
5353 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%  
5354 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%  
5355 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%  
5356 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%  
5357 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
5358 \renewcommand*{\glsxtrfullformat}[2]{%  
5359   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%  
5360   \ifglsxtrininsertinside\else##2\fi  
5361   \glsxtruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%  
5362 }%  
5363 \renewcommand*{\glsxtrfullplformat}[2]{%  
5364   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%  
5365   \ifglsxtrininsertinside\else##2\fi  
5366   \glsxtruserparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}{##1}%  
5367 }%  
5368 \renewcommand*{\Glsxtrfullformat}[2]{%  
5369   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%  
5370   \ifglsxtrininsertinside\else##2\fi  
5371   \glsxtruserparen{\glsfirstlongfont{\glsaccesslong{##1}}}{##1}%  
5372 }%  
5373 \renewcommand*{\Glsxtrfullplformat}[2]{%  
5374   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%  
5375   \ifglsxtrininsertinside\else##2\fi  
5376   \Glsxtruserparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}{##1}%  
5377 }%
```

```

5375 \ifglxtrinsertinside\else##2\fi
5376 \glxtruserparen{\glsfirstlongfont{\glaccesslongpl{##1}}}{##1}%
5377 }%
5378 }

```

-long-user-desc

```

5379 \newabbreviationstyle{short-long-user-desc}%
5380 {%
5381 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
5382 }%
5383 {%
5384 \GlsXtrUseAbbrStyleFmts{short-long-user}%
5385 }

```

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 `\glentryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glabbrvfont`. 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 `\glxtrshort` 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:

```

5386 \let\@glxtr@org@markright\markright

```

Redefine (grouping not added in case it interferes with the original code):

```

5387 \renewcommand*{\markright}[1]{%
5388 \glxtrmarkhook
5389 \@glxtr@org@markright{\@glxtrinmark#1\@glxtrnotinmark}%
5390 \glxtrrestoremarkhook

```

5391 }

\markboth Save original definition:

5392 \let\@glxstr@org@markboth\markboth

Redefine (grouping not added in case it interferes with the original code):

5393 \renewcommand*{\markboth}[2]{%

5394 \glxstrmarkhook

5395 \@glxstr@org@markboth

5396 {\@glxstrinmark#1\@glxstrnotinmark}%

5397 {\@glxstrinmark#2\@glxstrnotinmark}%

5398 \glxstrrestoremarkhook

5399 }

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

5400 \newcommand*{\glxstrRevertMarks}{%

5401 \let\markright\@glxstr@org@markright

5402 \let\markboth\@glxstr@org@markboth

5403 }

\glxstrinmark

5404 \newcommand*{\glxstrinmark}[2]{#2}

\@glxstrinmark

5405 \newrobustcmd*{\@glxstrinmark}{%

5406 \let\glxstrinmark\@firstoftwo

5407 }

glxstrnotinmark

5408 \newrobustcmd*{\@glxstrnotinmark}{%

5409 \let\glxstrinmark\@secondoftwo

5410 }

\glxstrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

5411 \newcommand*{\glxstrmarkhook}{%

Save current definitions:

5412 \let\@glxstr@org@MakeUppercase\MakeUppercase

5413 \let\@glxstr@org@glxstrtitleshort\glxstrtitleshort

5414 \let\@glxstr@org@glxstrtitleshortpl\glxstrtitleshortpl

5415 \let\@glxstr@org@Glsxtrtitleshort\Glsxtrtitleshort

5416 \let\@glxstr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl

5417 \let\@glxstr@org@glxstrtitletext\glxstrtitletext

5418 \let\@glxstr@org@Glsxtrtitletext\Glsxtrtitletext

5419 \let\@glxstr@org@glxstrtitleplural\glxstrtitleplural

5420 \let\@glxstr@org@Glsxtrtitleplural\Glsxtrtitleplural


```

5421 \let\@glxstr@org@glxstrtitlefirst\glxstrtitlefirst
5422 \let\@glxstr@org@Glsxstrtitlefirst\Glsxstrtitlefirst
5423 \let\@glxstr@org@glxstrtitlefirstplural\glxstrtitlefirstplural
5424 \let\@glxstr@org@Glsxstrtitlefirstplural\Glsxstrtitlefirstplural
5425 \let\@glxstr@org@glxstrtitlelong\glxstrtitlelong
5426 \let\@glxstr@org@glxstrtitlelongpl\glxstrtitlelongpl
5427 \let\@glxstr@org@Glsxstrtitlelong\Glsxstrtitlelong
5428 \let\@glxstr@org@Glsxstrtitlelongpl\Glsxstrtitlelongpl
5429 \let\@glxstr@org@glxstrtitlefull\glxstrtitlefull
5430 \let\@glxstr@org@glxstrtitlefullpl\glxstrtitlefullpl
5431 \let\@glxstr@org@Glsxstrtitlefull\Glsxstrtitlefull
5432 \let\@glxstr@org@Glsxstrtitlefullpl\Glsxstrtitlefullpl

```

New definitions

```

5433 \let\glxstrifinmark\@firstoftwo
5434 \let\MakeUppercase\MakeTextUppercase
5435 \let\glxstrtitleshort\glxstrheadshort
5436 \let\glxstrtitleshortpl\glxstrheadshortpl
5437 \let\Glsxstrtitleshort\Glsxstrheadshort
5438 \let\Glsxstrtitleshortpl\Glsxstrheadshortpl
5439 \let\glxstrtitletext\glxstrheadtext
5440 \let\Glsxstrtitletext\Glsxstrheadtext
5441 \let\glxstrtitleplural\glxstrheadplural
5442 \let\Glsxstrtitleplural\Glsxstrheadplural
5443 \let\glxstrtitlefirst\glxstrheadfirst
5444 \let\Glsxstrtitlefirst\Glsxstrheadfirst
5445 \let\glxstrtitlefirstplural\glxstrheadfirstplural
5446 \let\Glsxstrtitlefirstplural\Glsxstrheadfirstplural
5447 \let\glxstrtitlelong\glxstrheadlong
5448 \let\glxstrtitlelongpl\glxstrheadlongpl
5449 \let\Glsxstrtitlelong\Glsxstrheadlong
5450 \let\Glsxstrtitlelongpl\Glsxstrheadlongpl
5451 \let\glxstrtitlefull\glxstrheadfull
5452 \let\glxstrtitlefullpl\glxstrheadfullpl
5453 \let\Glsxstrtitlefull\Glsxstrheadfull
5454 \let\Glsxstrtitlefullpl\Glsxstrheadfullpl
5455 }

```

`\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.)

```

5456 \newcommand*{\glxstrrestoremarkhook}{%
5457 \let\glxstrifinmark\@secondoftwo
5458 \let\MakeUppercase\@glxstr@org@MakeUppercase
5459 \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
5460 \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
5461 \let\Glsxstrtitleshort\@glxstr@org@Glsxstrtitleshort
5462 \let\Glsxstrtitleshortpl\@glxstr@org@Glsxstrtitleshortpl

```

```

5463 \let\glxtrtitletext\@glxtr@org@glxtrtitletext
5464 \let\Glsxtrtitletext\@glxtr@org@Glsxtrtitletext
5465 \let\glxtrtitleplural\@glxtr@org@glxtrtitleplural
5466 \let\Glsxtrtitleplural\@glxtr@org@Glsxtrtitleplural
5467 \let\glxtrtitlefirst\@glxtr@org@glxtrtitlefirst
5468 \let\Glsxtrtitlefirst\@glxtr@org@Glsxtrtitlefirst
5469 \let\glxtrtitlefirstplural\@glxtr@org@glxtrtitlefirstplural
5470 \let\Glsxtrtitlefirstplural\@glxtr@org@Glsxtrtitlefirstplural
5471 \let\glxtrtitlelong\@glxtr@org@glxtrtitlelong
5472 \let\glxtrtitlelongpl\@glxtr@org@glxtrtitlelongpl
5473 \let\Glsxtrtitlelong\@glxtr@org@Glsxtrtitlelong
5474 \let\Glsxtrtitlelongpl\@glxtr@org@Glsxtrtitlelongpl
5475 \let\glxtrtitlefull\@glxtr@org@glxtrtitlefull
5476 \let\glxtrtitlefullpl\@glxtr@org@glxtrtitlefullpl
5477 \let\Glsxtrtitlefull\@glxtr@org@Glsxtrtitlefull
5478 \let\Glsxtrtitlefullpl\@glxtr@org@Glsxtrtitlefullpl
5479 }

```

Instead of using one document-wide conditional, use headuc attribute to determine whether or not to use the all upper case form.

glxtrheadshort Command used to display short form in the page header.

```

5480 \newcommand*{\glxtrheadshort}[1]{%
5481 \protect\NoCaseChange
5482 {%
5483 \gl@ifattribute{#1}{headuc}{true}%
5484 {%
5485 \GLSxtrshort[noindex,hyper=false]{#1}[]%
5486 }%
5487 {%
5488 \glxtrshort[noindex,hyper=false]{#1}[]%
5489 }%
5490 }%
5491 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

5492 \newrobustcmd*{\lsxtrtitleshort}[1]{%
5493 \glxtrshort[noindex,hyper=false]{#1}[]%
5494 }

```

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.

```

5495 \newcommand*{\glxtrheadshortpl}[1]{%
5496 \protect\NoCaseChange
5497 {%
5498 \gl@ifattribute{#1}{headuc}{true}%
5499 {%
5500 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%

```

```

5501 }%
5502 {%
5503     \glsxtrshortpl [noindex,hyper=false] {#1} []%
5504 }%
5505 }%
5506 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```

5507 \newrobustcmd*{\glsxtrtitleshortpl} [1] {%
5508     \glsxtrshortpl [noindex,hyper=false] {#1} []%
5509 }

```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```

5510 \newcommand*{\Glsxtrheadshort} [1] {%
5511     \protect\NoCaseChange
5512 {%
5513     \glsifattribute{#1}{headuc}{true}%
5514     {%
5515         \GLSxtrshort [noindex,hyper=false] {#1} []%
5516     }%
5517     {%
5518         \Glsxtrshort [noindex,hyper=false] {#1} []%
5519     }%
5520 }%
5521 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5522 \newrobustcmd*{\Glsxtrtitleshort} [1] {%
5523     \Glsxtrshort [noindex,hyper=false] {#1} []%
5524 }

```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```

5525 \newcommand*{\Glsxtrheadshortpl} [1] {%
5526     \protect\NoCaseChange
5527 {%
5528     \glsifattribute{#1}{headuc}{true}%
5529     {%
5530         \GLSxtrshortpl [noindex,hyper=false] {#1} []%
5531     }%
5532     {%
5533         \Glsxtrshortpl [noindex,hyper=false] {#1} []%
5534     }%
5535 }%
5536 }

```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
5537 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
5538   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
5539 }
```

`\glxtrheadtext` As above but for the text value.

```
5540 \newcommand*{\glxtrheadtext}[1]{%
5541   \protect\NoCaseChange
5542   {%
5543     \glsifattribute{#1}{headuc}{true}%
5544     {%
5545       \GLStext[noindex,hyper=false]{#1}[]%
5546     }%
5547     {%
5548       \glstext[noindex,hyper=false]{#1}[]%
5549     }%
5550   }%
5551 }
```

`glxtrtitletext` Command to display text value in section title and table of contents.

```
5552 \newrobustcmd*{\glxtrtitletext}[1]{%
5553   \glstext[noindex,hyper=false]{#1}[]%
5554 }
```

`\Glsxtrheadtext` First letter converted to upper case

```
5555 \newcommand*{\Glsxtrheadtext}[1]{%
5556   \protect\NoCaseChange
5557   {%
5558     \glsifattribute{#1}{headuc}{true}%
5559     {%
5560       \GLStext[noindex,hyper=false]{#1}[]%
5561     }%
5562     {%
5563       \Glstext[noindex,hyper=false]{#1}[]%
5564     }%
5565   }%
5566 }
```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```
5567 \newrobustcmd*{\Glsxtrtitletext}[1]{%
5568   \Glstext[noindex,hyper=false]{#1}[]%
5569 }
```

`lsxtrheadplural` As above but for the plural value.

```
5570 \newcommand*{\lsxtrheadplural}[1]{%
5571   \protect\NoCaseChange
```

```

5572 {%
5573   \glsifattribute{#1}{headuc}{true}%
5574   {%
5575     \GLSplural[noindex,hyper=false]{#1}[]%
5576   }%
5577   {%
5578     \glsplural[noindex,hyper=false]{#1}[]%
5579   }%
5580 }%
5581 }

```

sxtrtitleplural Command to display plural value in section title and table of contents.

```

5582 \newrobustcmd*{\glsxtrtitleplural}[1]{%
5583   \glsplural[noindex,hyper=false]{#1}[]%
5584 }

```

lsxtrheadplural Convert first letter to upper case.

```

5585 \newcommand*{\Glsxtrheadplural}[1]{%
5586   \protect\NoCaseChange
5587   {%
5588     \glsifattribute{#1}{headuc}{true}%
5589     {%
5590       \GLSplural[noindex,hyper=false]{#1}[]%
5591     }%
5592     {%
5593       \Glsplural[noindex,hyper=false]{#1}[]%
5594     }%
5595   }%
5596 }

```

sxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```

5597 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
5598   \Glsplural[noindex,hyper=false]{#1}[]%
5599 }

```

glsxtrheadfirst As above but for the first value.

```

5600 \newcommand*{\glsxtrheadfirst}[1]{%
5601   \protect\NoCaseChange
5602   {%
5603     \glsifattribute{#1}{headuc}{true}%
5604     {%
5605       \GLSfirst[noindex,hyper=false]{#1}[]%
5606     }%
5607     {%
5608       \glsfirst[noindex,hyper=false]{#1}[]%
5609     }%
5610   }%
5611 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```
5612 \newrobustcmd*{\lsxtrtitlefirst}[1]{%
5613   \glsfirst[noindex,hyper=false]{#1}[]%
5614 }
```

Glsxtrheadfirst First letter converted to upper case

```
5615 \newcommand*{\Glsxtrheadfirst}[1]{%
5616   \protect\NoCaseChange
5617   {%
5618     \glsifattribute{#1}{headuc}{true}%
5619     {%
5620       \GLSfirst[noindex,hyper=false]{#1}[]%
5621     }%
5622     {%
5623       \Glsfirst[noindex,hyper=false]{#1}[]%
5624     }%
5625   }%
5626 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```
5627 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
5628   \Glsfirst[noindex,hyper=false]{#1}[]%
5629 }
```

headfirstplural As above but for the firstplural value.

```
5630 \newcommand*{\glsxtrheadfirstplural}[1]{%
5631   \protect\NoCaseChange
5632   {%
5633     \glsifattribute{#1}{headuc}{true}%
5634     {%
5635       \GLSfirstplural[noindex,hyper=false]{#1}[]%
5636     }%
5637     {%
5638       \glsfirstplural[noindex,hyper=false]{#1}[]%
5639     }%
5640   }%
5641 }
```

itlefirstplural Command to display firstplural value in section title and table of contents.

```
5642 \newrobustcmd*{\glsxtritlefirstplural}[1]{%
5643   \glsfirstplural[noindex,hyper=false]{#1}[]%
5644 }
```

headfirstplural First letter converted to upper case

```
5645 \newcommand*{\Glsxtrheadfirstplural}[1]{%
5646   \protect\NoCaseChange
5647   {%
```

```

5648 \glsifattribute{#1}{headuc}{true}%
5649 {%
5650 \GLSfirstplural[noindex,hyper=false]{#1}[]%
5651 }%
5652 {%
5653 \GLSfirstplural[noindex,hyper=false]{#1}[]%
5654 }%
5655 }%
5656 }

```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```

5657 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
5658 \GLSfirstplural[noindex,hyper=false]{#1}[]%
5659 }

```

`\glxtrheadlong` Command used to display long form in the page header.

```

5660 \newcommand*{\glxtrheadlong}[1]{%
5661 \protect\NoCaseChange
5662 {%
5663 \glsifattribute{#1}{headuc}{true}%
5664 {%
5665 \GLSxtrlong[noindex,hyper=false]{#1}[]%
5666 }%
5667 {%
5668 \glxtrlong[noindex,hyper=false]{#1}[]%
5669 }%
5670 }%
5671 }

```

`\glxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```

5672 \newrobustcmd*{\glxtrtitlelong}[1]{%
5673 \glxtrlong[noindex,hyper=false]{#1}[]%
5674 }

```

`\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.

```

5675 \newcommand*{\glxtrheadlongpl}[1]{%
5676 \protect\NoCaseChange
5677 {%
5678 \glsifattribute{#1}{headuc}{true}%
5679 {%
5680 \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
5681 }%
5682 {%
5683 \glxtrlongpl[noindex,hyper=false]{#1}[]%
5684 }%

```

```
5685 }%
5686 }
```

sxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents.

```
5687 \newrobustcmd*{\glsxtrtitlelongpl}[1]{%
5688   \glsxtrlongpl[noindex,hyper=false]{#1}[]%
5689 }
```

\Glsxtrheadlong Command used to display long form in the page header with the first letter converted to upper case.

```
5690 \newcommand*{\Glsxtrheadlong}[1]{%
5691   \protect\NoCaseChange
5692   {%
5693     \glusifattribute{#1}{headuc}{true}%
5694     {%
5695       \GLSxtrlong[noindex,hyper=false]{#1}[]%
5696     }%
5697     {%
5698       \Glsxtrlong[noindex,hyper=false]{#1}[]%
5699     }%
5700   }%
5701 }
```

Glsxtrtitlelong Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
5702 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
5703   \Glsxtrlong[noindex,hyper=false]{#1}[]%
5704 }
```

lsxtrheadlongpl Command used to display plural long form in the page header with the first letter converted to upper case.

```
5705 \newcommand*{\Glsxtrheadlongpl}[1]{%
5706   \protect\NoCaseChange
5707   {%
5708     \glusifattribute{#1}{headuc}{true}%
5709     {%
5710       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
5711     }%
5712     {%
5713       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
5714     }%
5715   }%
5716 }
```

sxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
5717 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
5718   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
5719 }
```


`\glsxtrheadfull` Command used to display full form in the page header.

```
5720 \newcommand*{\glsxtrheadfull}[1]{%
5721   \protect\NoCaseChange
5722   {%
5723     \glsifattribute{#1}{headuc}{true}%
5724     {%
5725       \GLSxtrfull[noindex,hyper=false]{#1}[]%
5726     }%
5727     {%
5728       \glsxtrfull[noindex,hyper=false]{#1}[]%
5729     }%
5730   }%
5731 }
```

`glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents.

```
5732 \newrobustcmd*{\glsxtrtitlefull}[1]{%
5733   \glsxtrfull[noindex,hyper=false]{#1}[]%
5734 }
```

`lsxtrheadfullpl` Command used to display plural full form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrfullpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```
5735 \newcommand*{\glsxtrheadfullpl}[1]{%
5736   \protect\NoCaseChange
5737   {%
5738     \glsifattribute{#1}{headuc}{true}%
5739     {%
5740       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
5741     }%
5742     {%
5743       \glsxtrfullpl[noindex,hyper=false]{#1}[]%
5744     }%
5745   }%
5746 }
```

`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents.

```
5747 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
5748   \glsxtrfullpl[noindex,hyper=false]{#1}[]%
5749 }
```

`\Glsxtrheadfull` Command used to display full form in the page header with the first letter converted to upper case.

```
5750 \newcommand*{\Glsxtrheadfull}[1]{%
5751   \protect\NoCaseChange
5752   {%
5753     \glsifattribute{#1}{headuc}{true}%
5754     {%
5755       \GLSxtrfull[noindex,hyper=false]{#1}[]%
```

```

5756 }%
5757 {%
5758   \Glsxtrfull[noindex,hyper=false]{#1}[]%
5759 }%
5760 }%
5761 }

```

Glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5762 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
5763   \Glsxtrfull[noindex,hyper=false]{#1}[]%
5764 }

```

lSxtrheadfullpl Command used to display plural full form in the page header with the first letter converted to upper case.

```

5765 \newcommand*{\Glsxtrheadfullpl}[1]{%
5766   \protect\NoCaseChange
5767   {%
5768     \glsifattribute{#1}{headuc}{true}%
5769     {%
5770       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
5771     }%
5772     {%
5773       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
5774     }%
5775   }%
5776 }

```

sxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

5777 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
5778   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
5779 }

```

\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.

```

5780 \ifdef\texorpdfstring
5781 {
5782   \newcommand*{\glsfmtshort}[1]{%
5783     \texorpdfstring
5784       {\glsxtrtitleshort{#1}}%
5785       {\glsentryshort{#1}}%
5786   }
5787 }
5788 {
5789   \newcommand*{\glsfmtshort}[1]{%
5790     \glsxtrtitleshort{#1}}
5791 }

```

Similarly for the plural version.

`\glsfmtshortpl`

```
5792 \ifdef\texorpdfstring
5793 {
5794   \newcommand*\glsfmtshortpl[1]{%
5795     \texorpdfstring
5796       {\glsxtrtitleshortpl{#1}}%
5797       {\glsentryshortpl{#1}}%
5798   }
5799 }
5800 {
5801   \newcommand*\glsfmtshortpl[1]{%
5802     \glsxtrtitleshortpl{#1}}
5803 }
```

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).

```
5804 \ifdef\texorpdfstring
5805 {
5806   \newcommand*\Glsfmtshort[1]{%
5807     \texorpdfstring
5808       {\Glsxtrtitleshort{#1}}%
5809       {\glsentryshort{#1}}%
5810   }
5811 }
5812 {
5813   \newcommand*\Glsfmtshort[1]{%
5814     \Glsxtrtitleshort{#1}}
5815 }
```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```
5816 \ifdef\texorpdfstring
5817 {
5818   \newcommand*\Glsfmtshortpl[1]{%
5819     \texorpdfstring
5820       {\Glsxtrtitleshortpl{#1}}%
5821       {\glsentryshortpl{#1}}%
5822   }
5823 }
5824 {
5825   \newcommand*\Glsfmtshortpl[1]{%
5826     \Glsxtrtitleshortpl{#1}}
5827 }
```

`\glsfmttext` As above but for the text value.

```
5828 \ifdef\texorpdfstring
```

```

5829 {
5830   \newcommand*{\glsfmttext}[1]{%
5831     \texorpdfstring
5832     {\glsxtrtitletext{#1}}%
5833     {\glsentrytext{#1}}%
5834   }
5835 }
5836 {
5837   \newcommand*{\glsfmttext}[1]{%
5838     \glsxtrtitletext{#1}}
5839 }

```

`\Glsfmttext` First letter converted to upper case.

```

5840 \ifdef\texorpdfstring
5841 {
5842   \newcommand*{\Glsfmttext}[1]{%
5843     \texorpdfstring
5844     {\Glsxtrtitletext{#1}}%
5845     {\glsentrytext{#1}}%
5846   }
5847 }
5848 {
5849   \newcommand*{\Glsfmttext}[1]{%
5850     \Glsxtrtitletext{#1}}
5851 }

```

`\glsfmtplural` As above but for the plural value.

```

5852 \ifdef\texorpdfstring
5853 {
5854   \newcommand*{\glsfmtplural}[1]{%
5855     \texorpdfstring
5856     {\glsxtrtitleplural{#1}}%
5857     {\glsentryplural{#1}}%
5858   }
5859 }
5860 {
5861   \newcommand*{\glsfmtplural}[1]{%
5862     \glsxtrtitleplural{#1}}
5863 }

```

`\Glsfmtplural` First letter converted to upper case.

```

5864 \ifdef\texorpdfstring
5865 {
5866   \newcommand*{\Glsfmtplural}[1]{%
5867     \texorpdfstring
5868     {\Glsxtrtitleplural{#1}}%
5869     {\glsentryplural{#1}}%
5870   }
5871 }

```

```

5872 {
5873   \newcommand*{\Glsfmtplural}[1]{%
5874     \Glsxtrtitleplural{#1}}
5875 }

```

\glsfmtfirst As above but for the first value.

```

5876 \ifdef\texorpdfstring
5877 {
5878   \newcommand*{\glsfmtfirst}[1]{%
5879     \texorpdfstring
5880     {\Glsxtrtitlefirst{#1}}%
5881     {\Glsentryfirst{#1}}%
5882   }
5883 }
5884 {
5885   \newcommand*{\glsfmtfirst}[1]{%
5886     \Glsxtrtitlefirst{#1}}
5887 }

```

\Glsfmtfirst First letter converted to upper case.

```

5888 \ifdef\texorpdfstring
5889 {
5890   \newcommand*{\Glsfmtfirst}[1]{%
5891     \texorpdfstring
5892     {\Glsxtrtitlefirst{#1}}%
5893     {\Glsentryfirst{#1}}%
5894   }
5895 }
5896 {
5897   \newcommand*{\Glsfmtfirst}[1]{%
5898     \Glsxtrtitlefirst{#1}}
5899 }

```

\glsfmtfirstpl As above but for the firstplural value.

```

5900 \ifdef\texorpdfstring
5901 {
5902   \newcommand*{\glsfmtfirstpl}[1]{%
5903     \texorpdfstring
5904     {\Glsxtrtitlefirstplural{#1}}%
5905     {\Glsentryfirstplural{#1}}%
5906   }
5907 }
5908 {
5909   \newcommand*{\glsfmtfirstpl}[1]{%
5910     \Glsxtrtitlefirstplural{#1}}
5911 }

```

\Glsfmtfirstpl First letter converted to upper case.

```

5912 \ifdef\texorpdfstring

```

```

5913 {
5914   \newcommand*{\Glsfmtfirstpl}[1]{%
5915     \texorpdfstring
5916     {\Glsxtrtitlefirstplural{#1}}%
5917     {\glsentryfirstplural{#1}}%
5918   }
5919 }
5920 {
5921   \newcommand*{\Glsfmtfirstpl}[1]{%
5922     \Glsxtrtitlefirstplural{#1}}
5923 }

```

`\glsfmtlong` As above but for the long value.

```

5924 \ifdef\texorpdfstring
5925 {
5926   \newcommand*{\glsfmtlong}[1]{%
5927     \texorpdfstring
5928     {\glsxtrtitlelong{#1}}%
5929     {\glsentrylong{#1}}%
5930   }
5931 }
5932 {
5933   \newcommand*{\glsfmtlong}[1]{%
5934     \glsxtrtitlelong{#1}}
5935 }

```

`\Glsfmtlong` First letter converted to upper case.

```

5936 \ifdef\texorpdfstring
5937 {
5938   \newcommand*{\Glsfmtlong}[1]{%
5939     \texorpdfstring
5940     {\Glsxtrtitlelong{#1}}%
5941     {\glsentrylong{#1}}%
5942   }
5943 }
5944 {
5945   \newcommand*{\Glsfmtlong}[1]{%
5946     \Glsxtrtitlelong{#1}}
5947 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

5948 \ifdef\texorpdfstring
5949 {
5950   \newcommand*{\glsfmtlongpl}[1]{%
5951     \texorpdfstring
5952     {\glsxtrtitlelongpl{#1}}%
5953     {\glsentrylongpl{#1}}%
5954   }
5955 }

```

```

5956 {
5957   \newcommand*{\glsfmtlongpl}[1]{%
5958     \glsxtrtitlelongpl{#1}}
5959 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

5960 \ifdef\teorpdfstring
5961 {
5962   \newcommand*{\Glsfmtlongpl}[1]{%
5963     \teorpdfstring
5964     {\Glsxtrtitlelongpl{#1}}%
5965     {\glsentrylongpl{#1}}%
5966   }
5967 }
5968 {
5969   \newcommand*{\Glsfmtlongpl}[1]{%
5970     \Glsxtrtitlelongpl{#1}}
5971 }

```

`\glsfmtfull` In-line full format.

```

5972 \ifdef\teorpdfstring
5973 {
5974   \newcommand*{\glsfmtfull}[1]{%
5975     \teorpdfstring
5976     {\glsxtrtitlefull{#1}}%
5977     {\glsxtrinlinefullformat{#1}-{}}%
5978   }
5979 }
5980 {
5981   \newcommand*{\glsfmtfull}[1]{%
5982     \glsxtrtitlefull{#1}}
5983 }

```

`\Glsfmtfull` First letter converted to upper case.

```

5984 \ifdef\teorpdfstring
5985 {
5986   \newcommand*{\Glsfmtfull}[1]{%
5987     \teorpdfstring
5988     {\Glsxtrtitlefull{#1}}%
5989     {\Glsxtrinlinefullformat{#1}-{}}%
5990   }
5991 }
5992 {
5993   \newcommand*{\Glsfmtfull}[1]{%
5994     \Glsxtrtitlefull{#1}}
5995 }

```

`\glsfmtfullpl` In-line full plural format.

```

5996 \ifdef\teorpdfstring

```

```

5997 {
5998   \newcommand*{\glsfmtfullpl}[1]{%
5999     \texorpdfstring
6000     {\glsxtrtitlefullpl{#1}}%
6001     {\glsxtrinlinefullplformat{#1}{}}%
6002   }
6003 }
6004 {
6005   \newcommand*{\glsfmtfullpl}[1]{%
6006     \glsxtrtitlefullpl{#1}}
6007 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

6008 \ifdef\texorpdfstring
6009 {
6010   \newcommand*{\Glsfmtfullpl}[1]{%
6011     \texorpdfstring
6012     {\Glsxtrtitlefullpl{#1}}%
6013     {\Glsxtrinlinefullplformat{#1}{}}%
6014   }
6015 }
6016 {
6017   \newcommand*{\Glsfmtfullpl}[1]{%
6018     \Glsxtrtitlefullpl{#1}}
6019 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

6020 \newcommand*{\RequireGlossariesExtraLang}[1]{%
6021   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
6022 }

```

`sariesExtraLang`

```

6023 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
6024   \ProvidesFile{glossariesxtr-#1.ldf}%
6025 }

```

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.)

```

6026 \@ifpackageloaded{tracklang}
6027 {%
6028   \AnyTrackedLanguages

```



```

6029  {%
6030    \ForEachTrackedDialect{\this@dialect}{%
6031      \IfTrackedLanguageFileExists{\this@dialect}%
6032      {glossariesxtr-}% prefix
6033      {.ldf}%
6034      {%
6035        \RequireGlossariesExtraLang{\CurrentTrackedTag}%
6036      }%
6037      {%
6038      }%
6039    }%
6040  }%
6041  {}%
6042 }
6043 {}

```

Load glossaries-extra-stylemods if required.

```
6044 \@glsxtr@redefstyles
```

and set the style:

```
6045 \@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:

```
6046 \NeedsTeXFormat{LaTeX2e}
6047 \ProvidesPackage{glossaries-extra-stylemods}[2016/08/15 v1.07 (NLCT)]
```

Provide package options to automatically load required predefined styles. The simplest method is to just test for the existence of the file `glossary-⟨option⟩.sty`. Packages can't be loaded whilst the options are being processed, so save the list in `\@glxtr@loadstyles`.

`sxtr@loadstyles`

```
6048 \newcommand*{\@glxtr@loadstyles}{%
6049 \DeclareOption*{%
6050   \IfFileExists{glossary-\CurrentOption.sty}%
6051   {\eappto\@glxtr@loadstyles{%
6052     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
6053   {\PackageError{glossaries-extra-styles}%
6054     {Unknown option '\CurrentOption'}{}}
6055 }
```

Process the package options:

```
6056 \ProcessOptions
```

Load the required packages:

```
6057 \@glxtr@loadstyles
```

Adjust the styles that the post description hook added, but only for styles that have already been defined. All the tree styles in `glossary-tree` include the post description hook, so they don't require adjustment. Similarly for `glossary-mcols` which builds on the tree styles.

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

`ewglossarystyle`

```
6058 \providecommand{\renewglossarystyle}[2]{%
6059   \ifcsundef{@glstyle@#1}%
6060   {%
6061     \PackageError{glossaries}{Glossary style '#1' isn't already defined}{}%

```

```

6062 }%
6063 {%
6064   \csdef{@glsstyle@#1}{#2}%
6065 }%
6066 }

```

2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying.

```

6067 \ifdefined{@glsstyle@listdotted}
6068 {%
6069   \renewglossarystyle{listdotted}{%
6070     \setglossarystyle{list}%
6071     \renewcommand*{\glossentry}[2]{%
6072       \item[]\makebox[\glslistdottedwidth][l]{%
6073         \glsentryitem{##1}%
6074         \glstarget{##1}{\glossentryname{##1}}%
6075         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
6076         \glossentrydesc{##1}\glspostdescription}%
6077     \renewcommand*{\subglossentry}[3]{%
6078       \item[]\makebox[\glslistdottedwidth][l]{%
6079         \glssubentryitem{##2}%
6080         \glstarget{##2}{\glossentryname{##2}}%
6081         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
6082         \glossentrydesc{##2}\glspostdescription}%
6083   }
6084 }
6085 {}

```

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.

```

6086 \ifcsdef{@glsstyle@long3col}
6087 {%
6088   \renewglossarystyle{long3col}{%
6089     \renewenvironment{theglossary}%
6090       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
6091       {\end{longtable}}}%
6092   \renewcommand*{\glossaryheader}{}%
6093   \renewcommand*{\glsgroupheading}[1]{%
6094     \renewcommand{\glossentry}[2]{%
6095       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6096       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline

```

```

6097 }%
6098 \renewcommand{\subglossentry}[3]{%
6099     &
6100     \glssubentryitem{##2}%
6101     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6102     ##3\tabularnewline
6103 }%
6104 \renewcommand*\{ \glsgroupskip }{%
6105     \ifglsgnoglobskip\else & & \tabularnewline\fi}%
6106 }
6107 }
6108 {}

```

Four column style:

```

6109 \ifcsdef{@glsstyle@long4col}
6110 {%
6111     \renewglossarystyle{long4col}{%
6112         \renewenvironment{theglossary}%
6113             {\begin{longtable}{llll}}%
6114             {\end{longtable}}%
6115         \renewcommand*\{ \glossaryheader }{%
6116             \renewcommand*\{ \glsgroupheading }[1]{%
6117                 \renewcommand{\glossentry}[2]{%
6118                     \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6119                     \glossentrydesc{##1}\glspostdescription &
6120                     \glossentrysymbol{##1} &
6121                     ##2\tabularnewline
6122                 }%
6123                 \renewcommand{\subglossentry}[3]{%
6124                     &
6125                     \glssubentryitem{##2}%
6126                     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6127                     \glossentrysymbol{##2} & ##3\tabularnewline
6128                 }%
6129                 \renewcommand*\{ \glsgroupskip }{%
6130                     \ifglsgnoglobskip\else & & \tabularnewline\fi}%
6131             }
6132         }
6133     }

```

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.

```

6134 \ifcsdef{@glsstyle@longragged3col}
6135 {%
6136     \renewglossarystyle{longragged3col}{%

```

```

6137 \renewenvironment{theglossary}%
6138   {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}%
6139     >{\raggedright}p{\glspagelistwidth}}}%
6140   {\end{longtable}}}%
6141 \renewcommand*{\glossaryheader}{}%
6142 \renewcommand*{\glsgroupheading}[1]{}%
6143 \renewcommand{\glossentry}[2]{%
6144   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6145   \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
6146 }%
6147 \renewcommand{\subglossentry}[3]{%
6148   &
6149   \glssubentryitem{##2}%
6150   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6151   ##3\tabularnewline
6152 }%
6153 \renewcommand*{\glsgroupskip}{%
6154   \ifglsgnogroupskip\else & \tabularnewline\fi}%
6155 }
6156 }
6157 {}

```

Four column style:

```

6158 \ifcsdef{@glstyle@altlongragged4col}
6159 {%
6160   \renewglossarystyle{altlongragged4col}{%
6161     \renewenvironment{theglossary}%
6162       {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}l%
6163         >{\raggedright}p{\glspagelistwidth}}}%
6164       {\end{longtable}}}%
6165     \renewcommand*{\glossaryheader}{}%
6166     \renewcommand*{\glsgroupheading}[1]{}%
6167     \renewcommand{\glossentry}[2]{%
6168       \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6169       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
6170       ##2\tabularnewline
6171     }%
6172     \renewcommand{\subglossentry}[3]{%
6173       &
6174       \glssubentryitem{##2}%
6175       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6176       \glossentrysymbol{##2} & ##3\tabularnewline
6177     }%
6178     \renewcommand*{\glsgroupskip}{%
6179       \ifglsgnogroupskip\else & & \tabularnewline\fi}%
6180   }
6181 }
6182 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment, but not the two column styles.

```

6183 \ifcsdef{@glsstyle@super3col}
6184 {%
6185   \renewglossarystyle{super3col}{%
6186     \renewenvironment{theglossary}%
6187       {\tablehead{}\tabletail{}}%
6188       \begin{supertabular}{\lp{\glsdescwidth}p{\glspagelistwidth}}}%
6189       {\end{supertabular}}}%
6190   \renewcommand*{\glossaryheader}{}%
6191   \renewcommand*{\glsgroupheading}[1]{}%
6192   \renewcommand{\glossentry}[2]{%
6193     \glstryitem{##1}\glstarget{##1}{\glsentryname{##1}} &
6194     \glsentrydesc{##1}\glspostdescription & ##2\tabularnewline
6195   }%
6196   \renewcommand{\subglossentry}[3]{%
6197     &
6198     \glssubentryitem{##2}%
6199     \glstarget{##2}{\strut}\glsentrydesc{##2}\glspostdescription &
6200     ##3\tabularnewline
6201   }%
6202   \renewcommand*{\glsgroupskip}{%
6203     \ifglsnogroupskip\else & &\tabularnewline\fi}%
6204 }
6205 }
6206 {}

```

Four column styles:

```

6207 \ifcsdef{@glsstyle@super4col}
6208 {%
6209   \renewglossarystyle{super4col}{%
6210     \renewenvironment{theglossary}%
6211       {\tablehead{}\tabletail{}}%
6212       \begin{supertabular}{\llll}}}%
6213       \end{supertabular}}}%
6214   \renewcommand*{\glossaryheader}{}%
6215   \renewcommand*{\glsgroupheading}[1]{}%
6216   \renewcommand{\glossentry}[2]{%
6217     \glstryitem{##1}\glstarget{##1}{\glsentryname{##1}} &
6218     \glsentrydesc{##1}\glspostdescription &
6219     \glsentrysymbol{##1} & ##2\tabularnewline
6220   }%
6221   \renewcommand{\subglossentry}[3]{%
6222     &
6223     \glssubentryitem{##2}%
6224     \glstarget{##2}{\strut}\glsentrydesc{##2}\glspostdescription &
6225     \glsentrysymbol{##2} & ##3\tabularnewline
6226   }%
6227   \renewcommand*{\glsgroupskip}{%

```

```

6228     \ifglsnogroupskip\else & & \tabularnewline\fi}%
6229 }
6230 }
6231 {}

```

2.6 Super Ragged Styles

The three and four column styles require adjustment, but not the two column styles.

```

6232 \ifcsdef{@glsstyle@superragged3col}
6233 {%
6234   \renewglossarystyle{superragged3col}{%
6235     \renewenvironment{theglossary}%
6236       {\tablehead{}\tabletail}%
6237       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%
6238         >{\raggedright}p{\glspagelistwidth}}}%
6239       {\end{supertabular}}}%
6240   \renewcommand*{\glossaryheader}{}%
6241   \renewcommand*{\glsgroupheading}[1]{}%
6242   \renewcommand{\glossentry}[2]{%
6243     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6244     \glossentrydesc{##1}\glspostdescription &
6245     ##2\tabularnewline
6246   }%
6247   \renewcommand{\subglossentry}[3]{%
6248     &
6249     \glssubentryitem{##2}%
6250     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6251     ##3\tabularnewline
6252   }%
6253   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
6254     &\tabularnewline\fi}%
6255 }
6256 }
6257 {}

```

Four columns:

```

6258 \ifcsdef{@glsstyle@altsuperragged4col}
6259 {%
6260   \renewglossarystyle{altsuperragged4col}{%
6261     \renewenvironment{theglossary}%
6262       {\tablehead{}\tabletail}%
6263       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}1%
6264         >{\raggedright}p{\glspagelistwidth}}}%
6265       {\end{supertabular}}}%
6266   \renewcommand*{\glossaryheader}{}%
6267   \renewcommand{\glossentry}[2]{%
6268     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6269     \glossentrydesc{##1}\glspostdescription &
6270     \glossentrysymbol{##1} & ##2\tabularnewline

```

```

6271 }%
6272 \renewcommand{\subglossentry}[3]{%
6273     &
6274     \glssubentryitem{##2}%
6275     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
6276     \glossentrysymbol{##2} & ##3\tabularnewline
6277 }%
6278 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & &
6279     &\tabularnewline\fi}%
6280 }
6281 }
6282 {}

```

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`.

```

6283 \ifdef{\@glsstyle@inline}
6284 {%
6285     \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
        Just use \glxtrpostdescription instead of \glspostdescription.
6286     \renewcommand*{\glsinlinedescformat}[3]{%
6287         \space#1\glxtrpostdescription}
6288     \renewcommand*{\glsinlinesubdescformat}[3]{%
6289         #1\glxtrpostdescription}
6290 }
6291 {}

```

2.8 Tree Styles

The `alttree` style is redefined to make it easier to made minor adjustments.

```

6292 \ifdef{\@glsstyle@alttree}
6293 {%

```

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.

```

6294 \newcommand{\glxtraltrtreeSymbolDescLocation}[2]{%
6295     {%
6296         \let\par\glxtrAltTreePar

```



```

6297      \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
6298      \glossentrydesc{#1}\glspostdescription \space #2\par
6299  }%
6300  }

```

`\trAltTreeIndent` Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

6301  \newlength\glxstrAltTreeIndent

```

`\lsxtrAltTreePar` Multi-paragraph descriptions need to keep the hanging indent.

```

6302  \newcommand{\glxstrAltTreePar}{%
6303    @@par
6304    \glxstrAltTreeSetHangIndent
6305    \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
6306  }

```

`\symbolDescLocation` `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

6307  \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
6308    \glxstralttreeSymbolDescLocation{#2}{#3}%
6309  }

```

`\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.

```

6310  \newlength\glxstrtreeindent

```

`\glxstralttreeInit` User-level initialisation for the alttree style.

```

6311  \newcommand*{\glxstralttreeInit}{%
6312    \settowidth{\glxstrtreeindent}{\glstreenamfmt{\glsgetwidestname\space}}%
6313    \glxstrAltTreeIndent=\parindent
6314  }

```

`\eglissetwidest` The original `\glissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

6315  \newcommand*{\eglissetwidest}[2][0]{%
6316    \protected@csedef{@glswidestname\romannumeral#1}{#2}%
6317  }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```

6318  \newcommand*{\xglissetwidest}[2][0]{%
6319    \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
6320  }

```

`\lsgetwidestname` Provide a user-level macro to obtain the widest top-level name.

```

6321  \newcommand*{\lsgetwidestname}{\@glswidestname}

```

`etwidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```
6322 \newcommand*\glsgetwidestsubname}[1]{%
6323   \ifcsundef{@glswidestname\romannumeral#1}%
6324     {\@glswidestname}%
6325     {\csuse{@glswidestname\romannumeral#1}}%
6326 }
```

`estTopLevelName` CamelCase is easier for long command names. Provide a CamelCase synonym of `\glsfindwidesttoplevelname`

```
6327 \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.

```
6328 \newrobustcmd*\glsFindWidestUsedTopLevelName}[1][\@glo@types]{%
6329   \dimen@=0pt\relax
6330   \gls@tmplen=0pt\relax
6331   \foralllglossaries[#1]{\@gls@type}%
6332   {%
6333     \forglsentries[\@gls@type]{\@glo@label}%
6334     {%
6335       \ifglsused{\@glo@label}%
6336       {%
6337         \ifglshasparent{\@glo@label}%
6338         {}%
6339         {%
6340           \settowidth{\dimen@}%
6341             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
6342           \ifdim\dimen@>\gls@tmplen
6343             \gls@tmplen=\dimen@
6344             \eglssetwidest{\glsentryname{\@glo@label}}%
6345           \fi
6346         }%
6347       }%
6348     }%
6349   }%
6350 }%
6351 }
```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```
6352 \newrobustcmd*\glsFindWidestUsedAnyName}[1][\@glo@types]{%
6353   \dimen@=0pt\relax
6354   \gls@tmplen=0pt\relax
6355   \foralllglossaries[#1]{\@gls@type}%
6356   {%
6357     \forglsentries[\@gls@type]{\@glo@label}%
6358     {%
```

```

6359     \ifglsused{\@glo@label}%
6360     {%
6361         \settowidth{\dimen@}%
6362         {\glstreenamefmt{\glstentryname{\@glo@label}}}%
6363         \ifdim\dimen@>\gls@tmplen
6364             \gls@tmplen=\dimen@
6365             \eglssetwidest{\glstentryname{\@glo@label}}%
6366         \fi
6367     }%
6368     {%
6369 }%
6370 }%
6371 }

```

ndWidestAnyName Like the above but doesn't check if the entry has been used.

```

6372 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
6373     \dimen@=0pt\relax
6374     \gls@tmplen=0pt\relax
6375     \forallglossaries[#1]{\@gls@type}%
6376     {%
6377         \forglsentries[\@gls@type]{\@glo@label}%
6378         {%
6379             \settowidth{\dimen@}%
6380             {\glstreenamefmt{\glstentryname{\@glo@label}}}%
6381             \ifdim\dimen@>\gls@tmplen
6382                 \gls@tmplen=\dimen@
6383                 \eglssetwidest{\glstentryname{\@glo@label}}%
6384             \fi
6385         }%
6386     }%
6387 }

```

estUsedLevelTwo This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well.
Any entry that has a great-grandparent is ignored.

```

6388 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
6389     \dimen@=0pt\relax
6390     \dimen@i=0pt\relax
6391     \dimen@ii=0pt\relax
6392     \forallglossaries[#1]{\@gls@type}%
6393     {%
6394         \forglsentries[\@gls@type]{\@glo@label}%
6395         {%
6396             \ifglsused{\@glo@label}%
6397             {%
6398                 \ifglshasparent{\@glo@label}%
6399                 {%
6400                     \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
6401                     \ifglshasparent{\@glo@parent}%
6402                 }%

```

```

6403         \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
6404         \ifglshasparent{\@glo@parent}%
6405         {%
6406         {%
6407             \settowidth{\gls@tmplen}%
6408             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6409             \ifdim\gls@tmplen>\dimen@ii
6410             \dimen@ii=\gls@tmplen
6411             \eglssetwidest[2]{\glsentryname{\@glo@label}}%
6412             \fi
6413         }%
6414     }%
6415     {%
6416         \settowidth{\gls@tmplen}%
6417         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6418         \ifdim\gls@tmplen>\dimen@i
6419         \dimen@i=\gls@tmplen
6420         \eglssetwidest[1]{\glsentryname{\@glo@label}}%
6421         \fi
6422     }%
6423 }%
6424 {%
6425     \settowidth{\gls@tmplen}%
6426     {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6427     \ifdim\gls@tmplen>\dimen@
6428     \dimen@=\gls@tmplen
6429     \eglssetwidest{\glsentryname{\@glo@label}}%
6430     \fi
6431 }%
6432 }%
6433 {}%
6434 }%
6435 }%
6436 }

```

`dWidestLevelTwo` This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

6437 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
6438     \dimen@=0pt\relax
6439     \dimen@i=0pt\relax
6440     \dimen@ii=0pt\relax
6441     \foralllglossaries[#1]{\@gls@type}%
6442     {%
6443         \forglsentries[\@gls@type]{\@glo@label}%
6444         {%
6445             \ifglshasparent{\@glo@label}%
6446             {%
6447                 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
6448                 \ifglshasparent{\@glo@parent}%
6449                 {%

```

```

6450      \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
6451      \ifglshasparent{\@glo@parent}%
6452      {}%
6453      {%
6454          \settowidth{\gls@tmplen}%
6455              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6456          \ifdim\gls@tmplen>\dimen@ii
6457              \dimen@ii=\gls@tmplen
6458              \eglssetwidest[2]{\glsentryname{\@glo@label}}%
6459          \fi
6460      }%
6461  }%
6462  {%
6463      \settowidth{\gls@tmplen}%
6464          {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6465      \ifdim\gls@tmplen>\dimen@i
6466          \dimen@i=\gls@tmplen
6467          \eglssetwidest[1]{\glsentryname{\@glo@label}}%
6468      \fi
6469  }%
6470  }%
6471  {%
6472      \settowidth{\gls@tmplen}%
6473          {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6474      \ifdim\gls@tmplen>\dimen@
6475          \dimen@=\gls@tmplen
6476          \eglssetwidest{\glsentryname{\@glo@label}}%
6477      \fi
6478  }%
6479  }%
6480  }%
6481  }

```

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.

```

6482  \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
6483      \dimen@=0pt\relax
6484      \gls@tmplen=0pt\relax
6485      #2=0pt\relax
6486      \forallglossaries[#1]{\@gls@type}%
6487      {%
6488          \forglsentries[\@gls@type]{\@glo@label}%
6489          {%
6490              \ifglused{\@glo@label}%
6491              {%
6492                  \settowidth{\dimen@}%
6493                      {\glstreenamefmt{\glsentryname{\@glo@label}}}%
6494                  \ifdim\dimen@>\gls@tmplen
6495                      \gls@tmplen=\dimen@

```

```

6496         \eglssetwidest{\glsentryname{\@glo@label}}%
6497         \fi
6498         \settowidth{\dimen@}%
6499         {\glsentrysymbol{\@glo@label}}%
6500         \ifdim\dimen@>#2\relax
6501             #2=\dimen@
6502         \fi
6503     }%
6504     {}%
6505 }%
6506 }%
6507 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

6508 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
6509     \dimen@=0pt\relax
6510     \gls@tmplen=0pt\relax
6511     #2=0pt\relax
6512     \forallglossaries[#1]{\@gls@type}%
6513     {%
6514         \forglsentries[\@gls@type]{\@glo@label}%
6515         {%
6516             \settowidth{\dimen@}%
6517             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
6518             \ifdim\dimen@>\gls@tmplen
6519                 \gls@tmplen=\dimen@
6520                 \eglssetwidest{\glsentryname{\@glo@label}}%
6521             \fi
6522             \settowidth{\dimen@}%
6523             {\glsentrysymbol{\@glo@label}}%
6524             \ifdim\dimen@>#2\relax
6525                 #2=\dimen@
6526             \fi
6527         }%
6528     }%
6529 }

```

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.

```

6530 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
6531     \dimen@=0pt\relax
6532     \gls@tmplen=0pt\relax
6533     #2=0pt\relax
6534     #3=0pt\relax
6535     \forallglossaries[#1]{\@gls@type}%
6536     {%
6537         \forglsentries[\@gls@type]{\@glo@label}%

```

```

6538     {%
6539         \ifglsused{\@glo@label}%
6540     {%
6541         \settowidth{\dimen@}%
6542             {\glstreenamefmt{\glentryname{\@glo@label}}}%
6543         \ifdim\dimen@>\gls@tmplen
6544             \gls@tmplen=\dimen@
6545             \eglssetwidest{\glentryname{\@glo@label}}%
6546         \fi
6547         \settowidth{\dimen@}%
6548             {\glentrysymbol{\@glo@label}}%
6549         \ifdim\dimen@>#2\relax
6550             #2=\dimen@
6551         \fi
6552         \settowidth{\dimen@}%
6553             {\GlsXtrFormatLocationList{\glentrynumberlist{\@glo@label}}}%
6554         \ifdim\dimen@>#3\relax
6555             #3=\dimen@
6556         \fi
6557     }%
6558 }%
6559 }%
6560 }%
6561 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

6562 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
6563     \dimen@=0pt\relax
6564     \gls@tmplen=0pt\relax
6565     #2=0pt\relax
6566     #3=0pt\relax
6567     \foralllglossaries[#1]{\@gls@type}%
6568     {%
6569         \forglsentries[\@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         \settowidth{\dimen@}%
6578             {\glentrysymbol{\@glo@label}}%
6579         \ifdim\dimen@>#2\relax
6580             #2=\dimen@
6581         \fi
6582         \settowidth{\dimen@}%
6583             {\GlsXtrFormatLocationList{\glentrynumberlist{\@glo@label}}}%
6584         \ifdim\dimen@>#3\relax

```

```

6585         #3=\dimen@
6586     \fi
6587 }%
6588 }%
6589 }

```

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.

```

6590 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
6591     \dimen@=0pt\relax
6592     \gls@tmplen=0pt\relax
6593     #2=0pt\relax
6594     \forallglossaries[#1]{\@gls@type}%
6595     {%
6596         \forglstentries[\@gls@type]{\@glo@label}%
6597         {%
6598             \ifglstused{\@glo@label}%
6599             {%
6600                 \settowidth{\dimen@}%
6601                 {\glstreenamfmt{\glstentryname{\@glo@label}}}%
6602                 \ifdim\dimen@>\gls@tmplen
6603                     \gls@tmplen=\dimen@
6604                     \eglissetwidest{\glstentryname{\@glo@label}}%
6605                 \fi
6606                 \settowidth{\dimen@}%
6607                 {\GlsXtrFormatLocationList{\glstentrynumberlist{\@glo@label}}}%
6608                 \ifdim\dimen@>#2\relax
6609                     #2=\dimen@
6610                 \fi
6611             }%
6612         }%
6613     }%
6614 }%
6615 }

```

AnyNameLocation Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

6616 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
6617     \dimen@=0pt\relax
6618     \gls@tmplen=0pt\relax
6619     #2=0pt\relax
6620     \forallglossaries[#1]{\@gls@type}%
6621     {%
6622         \forglstentries[\@gls@type]{\@glo@label}%
6623         {%
6624             \settowidth{\dimen@}%
6625             {\glstreenamfmt{\glstentryname{\@glo@label}}}%
6626             \ifdim\dimen@>\gls@tmplen
6627                 \gls@tmplen=\dimen@

```



```

6628         \eglssetwidest{\glstryname{\@glo@label}}}%
6629     \fi
6630     \settowidth{\dimen@}%
6631     {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
6632     \ifdim\dimen@>#2\relax
6633         #2=\dimen@
6634     \fi
6635 }%
6636 }%
6637 }

```

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`.

```

6638 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
6639     \glstreeindent=\glxtrtreetopindent\relax
6640 }

```

teTreeSubIndent

```

6641 %\cs{\glsxtrComputeTreeSubIndent}\marg{level}\marg{label}\marg{register}
6642 %\end{macrocode}
6643 % Compute the indent for the sub-entries. The first argument is the
6644 % level, the second argument is the entry label and the third
6645 % argument is the length register used to store the computed indent.
6646 % \begin{macrocode}
6647 \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
6648     \ifcsundef{@glswidestname\romannumeral#1}%
6649     {%
6650         \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
6651     }%
6652     {%
6653         \settowidth{#3}{\glstreenamefmt{%
6654             \csname @glswidestname\romannumeral#1\endcsname\space}}%
6655     }%
6656 }

```

eeSetHangIndent Set `\hangindent` for top-level entries:

```

6657 \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set `\hangindent` for sub-entries:

```

6658 \newcommand*{\glsxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine `alttree`:

```

6659 \renewglossarystyle{alttree}{%
6660     \renewenvironment{theglossary}%
6661     {%
6662         \glxtralttreeInit
6663         \def\@gls@prevlevel{-1}%

```

```

6664     \mbox{}\par}%
6665     {\par}%
6666 \renewcommand*\glossaryheader{}%
6667 \renewcommand*\glsgroupheading[1]{}%
6668 \renewcommand\glossentry[2]{%
6669     \ifnum\@gls@prevlevel=0\relax
6670     \else
6671         \glstrComputeTreeIndent{##1}%
6672     \fi
6673     \parindent\glstreeindent
6674     \glstrAltTreeSetHangIndent
6675     \makebox[Opt][r]%
6676     {%
6677         \glstreenamebox{\glstreeindent}%
6678         {%
6679             \glsentryitem{##1}%
6680             \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
6681         }%
6682     }%
6683     \glstralttreeSymbolDescLocation{##1}{##2}%
6684     \def\@gls@prevlevel{0}%
6685 }
6686 \renewcommand\subglossentry[3]{%
6687     \ifnum##1=1\relax
6688         \glssubentryitem{##2}%
6689     \fi
6690     \ifnum\@gls@prevlevel=##1\relax
6691     \else
6692         \glstrComputeTreeSubIndent{##1}{##2}{\@gls@tmplen}%
6693         \ifnum\@gls@prevlevel<##1\relax
6694             \setlength\glstreeindent\@gls@tmplen
6695             \addtolength\glstreeindent\parindent
6696             \parindent\glstreeindent
6697         \else
6698             \ifnum\@gls@prevlevel=0\relax
6699                 \glstrComputeTreeIndent{##2}%
6700             \else
6701                 \glstrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
6702             \fi
6703             \addtolength\parindent{-\glstreeindent}%
6704             \setlength\glstreeindent\parindent
6705         \fi
6706     \fi
6707     \glstrAltTreeSetSubHangIndent{##1}%
6708     \makebox[Opt][r]{\glstreenamebox{\@gls@tmplen}{%
6709         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
6710     \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
6711     \def\@gls@prevlevel{##1}%
6712 }%

```

```

6713 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
6714 }
6715 }%
6716 {%

```

Assume the style isn't required if it hasn't already been defined.

```

6717 }

```

Reset the default style

```

6718 \ifx\@glossary@default@style\relax
6719 \else
6720 \setglossarystyle{\@glxtr@current@style}
6721 \fi

```

Glossary

First use The first time a glossary entry is used (from the start of the document or after a reset) with one of the following commands: `\gls`, `\Gls`, `\GLS`, `\glspl`, `\Glspl`, `\GLSpl` or `\glsdisp`. *see* **first use flag** & **first use text**

First use flag A conditional that determines whether or not the entry has been used according to the rules of **first use**.

First use text The text that is displayed on **first use**, which is governed by the first and first-plural keys of `\newglossaryentry`. (May be overridden by `\glsdisp`.)

`makeindex` An indexing application.

`xindy` An flexible indexing application with multilingual support written in Perl.

Change History

0.1 (2015-11-22)		\@Glsymbol@: added redefinition	30
General: Initial experimental release	4	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	30
\Glsfmtshort: new	171	\@Glstext@: added redefinition	26
\glsfmtshort: new	170	\@Glsuseri@: added redefinition	31
\Glsfmtshortpl: new	171	\@Glsuserii@: added redefinition	31
\glsfmtshortpl: new	171	\@Glsuseriii@: added redefinition	32
short: switched inline full form to short		\@Glsuseriv@: added redefinition	32
(long)	137	\@Glsuserv@: added redefinition	32
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	32
\@ACRlong: added redefinition	36	\@acrlong: added redefinition	35
\@ACRlongpl: added redefinition	37	\@acrlongpl: added redefinition	36
\@ACRshort: added redefinition	34	\@acrshort: added redefinition	33
\@ACRshortpl: added redefinition	35	\@acrshortpl: added redefinition	34
\@Acrlong: added redefinition	35	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	36	argument	25
\@Acrshort: added redefinition	33	\@glsdescplural@: added redefinition	29
\@Acrshortpl: added redefinition	34	\@glsfirst@: added redefinition	26
\@GLSdesc@: added redefinition	29	\@glsfirstplural@: added redefinition	28
\@GLSdescplural@: added redefinition	30	\@glsplural@: added redefinition	27
\@GLSfirst@: added redefinition	27	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	28	redefinition	30
\@GLSname@: added redefinition	29	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	27	new	75
\@GLSSymbol@: added redefinition	30	\@glsxtr@field@linkdefs: new	25
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	110
redefinition	31	\@print@glossary: added redefinition	71
\@GLStext@: added redefinition	26	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	31	\abbrvdefaultfont	114
\@GLSuserii@: added redefinition	31	\glsaccessdesc: new	80
\@GLSuseriii@: added redefinition	32	\glsaccessdescplural: new	81
\@GLSuseriv@: added redefinition	32	\glsaccessfirst: new	78
\@GLSuserv@: added redefinition	32	\glsaccessfirstplural: new	78
\@GLSuservi@: added redefinition	33	\Glsaccesslong: new	82
\@Glsdesc@: added redefinition	29	\glsaccesslong: new	82
\@Glsdescplural@: added redefinition	29	\glsaccessname: new	76
\@Glsfirst@: added redefinition	26	\glsaccessplural: new	77
\@Glsfirstplural@: added redefinition	28	\Glsaccessshort: new	81
\@Glsname@: added redefinition	28	\glsaccessshort: new	81
\@Glsplural@: added redefinition	27	\Glsaccessshortpl: new	82

\glsaccesssshortpl: new	82	\cGLSpl: new	57
\glsaccessssymbol: new	79	\cGLSpl@: new	57
\glsaccessssymbolplural: new	79	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	76	new	53
\glstentryfmt: added check for short ..	24	\cGLS: new	57
\glslongpltok: new	110	\cGLSformat: new	57
\glsshortpltok: new	110	\cGLSpl: new	57
\glsxtrdiscardperiod: added check		\cGLSplformat: new	58
for plural	107	\GlossariesExtraWarningNoLine:	
\GLSxtrlongpl: new	123	new	6
\Glsxtrlongpl: new	123	\glsenableentrycount: new	53
\glsxtrlongpl: new	122	\glsfirstabbrvdefaultfont: new ..	114
\glsxtrNoGlossaryWarning: new	9	\glsfirstlongdefaultfont: new ...	114
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirst: new	173
new	106	\glsfmtfirst: new	173
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtfirstpl: new	173
new	106	\glsfmtfirstpl: new	173
\glsxtrpostlinkendsentence: new ..	106	\Glsfmtplural: new	172
\GLSxtrshortpl: new	121	\glsfmtplural: new	172
\Glsxtrshortpl: new	121	\Glsfmtshort: changed to use	
\glsxtrshortpl: new	120	\Glsxtrtitleshort	171
short-long-desc: fixed name to use		renamed from \Glstentryfmtshort ..	171
\glslabeltok	132	\glsfmtshort: changed to use	
\newabbreviation: fixed family name in		\glsxtrtitleshort	170
\setkeys	110	renamed from \glstentryfmtshort ..	170
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	130	\Glsxtrtitleshortpl	171
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl	171
redefinition of \acronymtype	7	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	171
\Glsxtrshort	171	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl	171
\glsxtrshort	170	\Glsfmttext: new	172
\Glsfmtshortpl: changed to use		\glsfmttext: new	171
\glsxtrshortpl	171	\glshasattribute: new	88
\glsfmtshortpl: changed to use		\glshascategoryattribute: new	87
\glsxtrshortpl	171	\GlsXtrEnableEntryCounting: new ..	52
\glsxtrifemptyglossary: new	12	\glsxtrifcounttrigger: new	55
\glsxtrnewnumber: added extra		\glsxtrscfont: new	142
argument	91	\glsxtrscsuffix: new	142
\glsxtrnewsymbol: added extra		\glsxtrsmfont: new	146
argument	91	\glsxtrsmsuffix: new	146
\MakeAcronymsAbbreviations: set the		short-em: new	152
default type to \acronymtype	66	short-em-desc: new	153
\newterm: fixed name argument	90	short-em-footnote: new	155
0.5 (2015-12-07)		short-em-long: new	151
\cGLS: new	57	short-em-long-desc: new	151
\cGLS@: new	57	short-em-postfootnote: new	155

short-sc-footnote: new	145	\Glsxtrheadtext: now uses headuc	
short-sc-postfootnote: new	145	attribute	164
short-sm: new	147	\glxtrheadtext: now uses headuc	
short-sm-desc: new	147	attribute	164
short-sm-footnote: new	148	short-long: switch off regular attribute	
short-sm-long: new	146	if set	131
short-sm-long-desc: new	147	short-long-desc: switch off regular	
short-sm-postfootnote: new	149	attribute if set	133
long-noshort-em: new	153	long-short: switch off regular attribute	
long-noshort-em-desc: new	154	if set	129
long-noshort-sm: new	148	long-short-desc: switch off regular	
long-noshort-sm-desc: new	148	attribute if set	131
long-short-em: new	149	footnote: switch off regular attribute if	
long-short-em-desc: new	150	set	133
long-short-sm: new	146	postfootnote: switch off regular	
long-short-sm-desc: new	146	attribute if set	135
0.5.1 (2015-12-02)		0.5.2 (2015-12-08)	
\Glsaccesstext: new	77	\@GLSdesc@: added accessibility support	29
0.5.1 (2015-12-07)		\@GLSdescplural@: added accessibility	
\@glxtr@doaccsupp: new	9	support	30
General: removed \ifglxtruseuhead	162	\@GLSfirst@: added accessibility	
\Glsaccessdesc: new	80	support	27
\Glsaccessdescplural: new	81	\@GLSfirstplural@: added accessibility	
\Glsaccessfirst: new	78	support	28
\Glsaccessfirstplural: new	78	\@GLSname@: added accessibility support	29
\Glsaccessname: new	76	\@GLSplural@: added accessibility	
\Glsaccessplural: new	77	support	27
\Glsaccesssymbol: new	79	\@GLSsymbol@: added accessibility	
\Glsaccesssymbolplural: new	80	support	30
\Glsxtrheadfirst: now uses headuc		\@GLSsymbolplural@: added	
attribute	166	accessibility support	31
\glxtrheadfirst: now uses headuc		\@GLStext@: added accessibility support	26
attribute	165	\@GLSdesc@: added accessibility support	29
\Glsxtrheadfirstplural: now uses		\@GLSdescplural@: added accessibility	
headuc attribute	166	support	29
\glxtrheadfirstplural: now uses		\@GLSfirst@: added accessibility	
headuc attribute	166	support	26
\Glsxtrheadplural: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	165	support	28
\glxtrheadplural: now uses headuc		\@GLSname@: add accessibility support ..	28
attribute	164	\@GLsplural@: added accessibility	
\Glsxtrheadshort: now uses headuc		support	27
attribute	163	\@GLSsymbol@: added accessibility	
\glxtrheadshort: now uses headuc		support	30
attribute	162	\@GLSsymbolplural@: added	
\Glsxtrheadshortpl: now uses headuc		accessibility support	30
attribute	163	\@GLStext@: added accessibility support	26
\glxtrheadshortpl: now uses headuc		\@GLSdesc@: added accessibility support	29
attribute	162		

\@glsdescplural@: added accessibility support	29	\glsxtrnewabbrevpresetkeyhook: new	112
\@glsfirst@: added accessibility support	26	\glsxtrtagfont: new	105
\@glsfirstplural@: added accessibility support	28	\KV@printgloss@nonumberlist: added	24
\@glsname@: added accessibility support	28	\mfu@checkword@do: added	104
\@glsplural@: added accessibility support	27	\setabbreviationstyle: added check for post-definition style switch	126
\@glsymbol@: added accessibility support	30	0.5.3 (2015-12-09)	
\@glsymbolplural@: added accessibility support	30	\@glsxtr@autoindex@at: new	100
\@glstext@: added accessibility support	26	\@glsxtr@autoindex@encap: new ...	101
\@glsxtr@activate@initialtagging: new	104	\@glsxtr@autoindex@esc: new	101
\@glsxtr@do@titlecaps@warn: new	104	\@glsxtr@autoindex@level: new ...	101
\@glsxtr@tag: new	104	\@glsxtr@autoindex@setname: new ..	99
General: fixed typo in glossaries-accsupp and tidied up code to use just one		\@glsxtr@doabbreviationsdef: new ..	7
\@ifpackageloaded	76	General: removed	
removed \glsxtrabbrvfmt	124	\GlsXtrNoGlsWarningNoAutoMakeMain	74
\glossaryentrynumbers: added	22	\glsdescwidth: added	21
\Glossentrydesc: added	102	\glspagelistwidth: added	21
\Glossentryname: added	96	\glsxtrdoautoindexname: new	99
\Glossentrysymbol: added	103	\glsxtrpostnamehook: new	98
\glossentrysymbol: added	102	\if@glsxtr@format@override: new ..	98
\GLSaccessdesc: new	80, 85	\ProvidesGlossariesExtraLang: new	176
\GLSaccessdescplural: new	81, 85	\RequireGlossariesExtraLang: new	176
\GLSaccessfirst: new	78, 84	0.5.4 (2015-12-15)	
\GLSaccessfirstplural: new	79, 84	\@newglossaryentry@defunitcounters: new	58
\GLSaccesslong: new	82, 86	\@GLSxtr@p@acrlong@: new	46
\GLSaccesslongpl: new	83, 86	\@GLSxtr@p@acrlongpl@: new	46
\Glsaccesslongpl: new	83	\@GLSxtr@p@acrshort@: new	45
\glsaccesslongpl: new	83	\@GLSxtr@p@acrshortpl@: new	46
\GLSaccessname: new	76, 83	\@GLSxtr@p@long@: new	45
\GLSaccessplural: new	77, 84	\@GLSxtr@p@longpl@: new	45
\GLSaccessshort: new	81, 86	\@GLSxtr@p@plural@: new	44
\GLSaccessshortpl: new	82, 86	\@GLSxtr@p@short@: new	44
\GLSaccesssymbol: new	79, 85	\@GLSxtr@p@shortpl@: new	45
\GLSaccesssymbolplural: new ...	80, 85	\@GLSxtr@p@text@: new	44
\GLSaccessstext: new	77, 84	\@GlsXtrEnableOnTheFly: new	17
\glsentryfmt: moved		\@Glsxtr: new	18
\glssetabbrvfmt from		\@Glsxtr@p@acrlong@: new	46
\glsxtrabbrvfmt to here	24	\@Glsxtr@p@acrlongpl@: new	46
\GlsXtrEnableInitialTagging: new	103	\@Glsxtr@p@acrshort@: new	45
\glsxtrfieldtitlecase: new	92	\@Glsxtr@p@acrshortpl@: new	45
\GlsXtrFormatLocationList: new ...	22	\@Glsxtr@p@long@: new	45
		\@Glsxtr@p@longpl@: new	45
		\@Glsxtr@p@plural@: new	44
		\@Glsxtr@p@short@: new	44
		\@Glsxtr@p@shortpl@: new	44
		\@Glsxtr@p@text@: new	43

\@Glsxtrpl: new	19	\glxtr: new	18
\@alt@glshyp@opt: new	42	\glxtrcat: new	18
\@glscalt@hyp@opt: new	41	\glxtrdowrglossaryhook: new	41
\@glscalt@hyp@opt@char: new	42	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	42	new	64
\@glscincrement@currunitcount:		\GlsXtrEnableOnTheFly: new	17
new	59	\Glsxtrpl: new	19
\@glsclocal@increment@currunitcount:		\glxtrpl: new	18
new	60	\glxtrpostlocalreset: new	52
\@glscsetdefault@glslink@opts:		\glxtrpostlocalunset: new	52
new	40	\glxtrpostreset: new	52
\@glxtr: new	18	\glxtrpostunset: new	51
\@glxtr@addunitcounter: new	59	\glxtrprotectlinks: new	43
\@glxtr@currunitcount: new	60	\GlsXtrSetAltModifier: new	42
\@glxtr@ifunitcounter: new	59	\GlsXtrSetDefaultGlsOpts: new	40
\@glxtr@p@acrlong@: new	46	\glxtrstarflywarn: new	17
\@glxtr@p@acrlongpl@: new	46	\GlsXtrWarning: new	19
\@glxtr@p@acrshort@: new	45	\MakeAcronymsAbbreviations: now	
\@glxtr@p@acrshortpl@: new	45	disables \setacronymstyle	66
\@glxtr@p@long@: new	45	1.0 (2016-01-24)	
\@glxtr@p@longpl@: new	45	\@glxtr@autoindexcrossrefs: new ..	6
\@glxtr@p@plural@: new	44	\@glxtr@idx@displaynumberlist:	
\@glxtr@p@short@: new	44	new	70
\@glxtr@p@shortpl@: new	44	\@glxtr@idx@entrynumberlist: new	71
\@glxtr@p@text@: new	43	\@glxtr@noidx@displaynumberlist:	
\@glxtr@prevunitcount: new	60	new	70
\@glxtr@setentryunitcountunsetattr:		\@glxtr@noidx@entrynumberlist:	
new	64	new	71
\@glxtr@unitcountlist: new	59	\@glxtr@noidx@numberlistloop:	
\@glxtrpl: new	18	new	70
\@newglossaryentryposthook: added		\@glxtr@reg@glosslist: new	67
empty see value if not set and added		\makeglossaries: new	67
‘see’ to field key map	14	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	17	\glxtrdiscardperiod: added check	
\cGlsformat: added	58	for first use	107
\cGlsformat: added	58	short-desc: fixed typo in	
\cGlsplformat: added	58	\glxtrinlinefullformat and	
\cGlsplformat: added	58	added missing second argument ...	138
\glsdisablehyper: added	42	1.02 (2016-04-25)	
\glsdohyperlink: added	42	\@glxtr@current@style: new	20
\glsdonohyperlink: added	42	\Glsfmtfull: new	175
\glsenableentryunitcount: new	60	\glsfmtfull: new	175
\glshasattribute: added check for		\Glsfmtfullpl: new	176
entry’s existence	88	\glsfmtfullpl: new	175
\glusifattribute: added check for		\Glsfmtlong: new	174
entry’s existence	88	\glsfmtlong: new	174
\glspostlinkhook: added existence		\Glsfmtlongpl: new	175
check	106	\glsfmtlongpl: new	174
\Glsxtr: new	18	\Glsxtrheadfull: new	169

\glxtrheadfull: new	169	\@GLSplural@: set abbreviation and regular format	27
\Glsxtrheadfullpl: new	170	\@GLSsymbol@: set regular format	30
\glxtrheadfullpl: new	169	\@GLSsymbolplural@: set regular format	31
\glxtrheadlong: new	168	\@GLStext@: set abbreviation and regular format	26
\glxtrheadlong: new	167	\@GLSuseri@: set regular format	31
\Glsxtrheadlongpl: new	168	\@GLSuserii@: set regular format	31
\glxtrheadlongpl: new	167	\@GLSuseriii@: set regular format	32
\Glsxtrtitlefull: new	170	\@GLSuseriv@: set regular format	32
\glxtrtitlefull: new	169	\@GLSuseriv@: set regular format	32
\Glsxtrtitlefullpl: new	170	\@GLSuservi@: set regular format	33
\glxtrtitlefullpl: new	169	\@Glsdesc@: set abbreviation and regular format	29
\Glsxtrtitlelong: new	168	\@Glsdescplural@: set abbreviation and regular format	29
\glxtrtitlelong: new	167	\@Glsfirst@: set abbreviation and regular format	26
\Glsxtrtitlelongpl: new	168	\@Glsfirstplural@: set abbreviation and regular format	28
\glxtrtitlelongpl: new	168	\@Glsname@: set abbreviation and regular format	28
\ifglxtrinsetinside: new	129	\@Glsplural@: set abbreviation and regular format	27
postfootnote: added redef of \glxtrsetupfulldefs	136	\@Glsymbol@: set regular format	30
stylemods: new	10	\@Glsymbolplural@: set regular format	30
1.03 (2016-04-27)		\@Glstext@: set abbreviation and regular format	26
\@GLSfirstplural@: bug fix: misspelt cs name	28	\@Glsuseri@: set regular format	31
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	27	\@Glsuserii@: set regular format	31
\@Glsfirstplural@: bug fix: misspelt cs name	28	\@Glsuseriii@: set regular format	32
\@Glsplural@: fixed bug \@Glsplural@ should be redefined not \@Glsplural	27	\@Glsuseriv@: set regular format	32
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	27	\@Glsuseriv@: set regular format	32
\glxtrtitlelongpl: bug fix: changed \glxtrlong to \glxtrlongpl ..	168	\@Glsuservi@: set regular format	32
\glxtrtitleshortpl: bug fix: changed \glxtrshort to \glxtrshortpl	163	\@gls@preglossaryhook: added check for entry's existence	105
1.04 (2015-04-30)		\@glsdesc@: set abbreviation and regular format	29
short-em-footnote: renamed from “footnote-em”	155	\@glsdescplural@: set abbreviation and regular format	29
1.04 (2016-05-02)		\@glsfirst@: set abbreviation and regular format	26
\@@glxtrpostloctag: new	23	\@glsfirstplural@: set abbreviation and regular format	28
\@GLSdesc@: set abbreviation and regular format	29	\@glsname@: set abbreviation and regular format	28
\@GLSdescplural@: set abbreviation and regular format	30	\@glsplural@: set abbreviation and regular format	27
\@GLSfirst@: set abbreviation format ..	27	\@glsymbol@: set regular format	30
\@GLSfirstplural@: set abbreviation and regular format	28		
\@GLSname@: set abbreviation and regular format	29		

\@glssymbolplural@: set regular format	30	short-em-nolong-desc: new	153
\@glstext@: set abbreviation and regular format	26	short-em-postfootnote: renamed from “postfootnote-em”	155
\@glstr@deprecated@abbrstyle:		short-footnote: new	135
new	128	short-long-user: new	158
\@glstr@do@style: new	11	short-long-user-desc: new	159
\@glstr@doloctag: new	24	short-nolong: new	138
\@glstr@idx@entrynumberlist:		short-nolong-desc: new	140
switched from \let to \newcommand	71	short-postfootnote: new	137
\@glstr@pagetag: new	23	short-sc-footnote: renamed from “footnote-sc”	145
\@glstr@pagetag: new	23	short-sc-nolong: new	143
\@glstr@preloctag: new	24	short-sc-nolong-desc: new	144
\@glstr@postloctag: new	23	short-sc-postfootnote: renamed from “postfootnote-sc”	145
\@glstr@preloctag: new	23	short-sm-footnote: renamed from “footnote-sm”	148
\glossentrydesc: added glossdescfont attribute check	92	short-sm-nolong: new	147
\Glossentryname: added glossnamefont attribute check	97	short-sm-nolong-desc: new	148
\glossentryname: added glossnamefont attribute check	94	short-sm-postfootnote: renamed from “postfootnote-sm”	149
moved post name hook inside condition	96	\letabbreviationstyle: new	128
\glsabbrvfont: new	149	\newabbreviationstyle: bug fix: corrected test for existence	127
\glsabbrvuserfont: new	156	long-em-noshort-em: new	153
\glsfirstabbrvfont: new	149	long-em-noshort-em-desc: new	154
\glsfirstabbrvuserfont: new	156	long-em-short-em: new	150
\glsfirstlongemfont: new	149	long-em-short-em-desc: new	151
\glsfirstlonguserfont: new	156	long-noshort: new	141
\glsifnotregularcategory: new	89	long-noshort-desc: new	141
\glslongdefaultfont: new	114	long-noshort-em: renamed from “long-em”	153
\glslongemfont: new	149	long-noshort-em-desc: renamed from “long-desc-em”	154
\glslongfont: new	114	long-noshort-sc: renamed from “long-sc”	144
\glslonguserfont: new	156	long-noshort-sc-desc: renamed from “long-desc-sc”	144
\glstrassignfieldfont: new	25	long-noshort-sm: renamed from “long-sm”	148
\GlsXtrEnablePreLocationTag: new	22	long-noshort-sm-desc: renamed from \long-desc-sm	148
\glstrfirstscfont: new	142	long-short-user: new	156
\glstrfirstsmfont: new	146	long-short-user-desc: new	157
\glstrlongshortdescsort: new	130	\renewabbreviationstyle: new style: new	11
\glstrpostnamehook: added category check	98		
\glstrregularfont: new	24	1.05 (2016-06-10)	
\glstruserfield: new	155	\eglssetwidest: new	185
\glstruserparen: new	155	\glsFindWidestAnyName: new	187
\glstrusersuffix: new	156		
\GlsXtrWarnDeprecatedAbbrStyle:			
new	128		
short-em-long-em: new	151		
short-em-long-em-desc: new	152		
short-em-nolong: new	153		

\glsFindWidestAnyNameLocation: new	192	\@GLSfirst@: added check for nohyperfirst attribute	27
\glsFindWidestAnyNameSymbol: new	190	\@GLSfirstplural@: added check for nohyperfirst attribute	28
\glsFindWidestAnyNameSymbolLocation: new	191	\@GLSxtrp: new	47
\glsFindWidestLevelTwo: new	188	\@Glsfirst@: added check for nohyperfirst attribute	27
\glsFindWidestUsedAnyName: new	186	\@Glsfirstplural@: added check for nohyperfirst attribute	28
\glsFindWidestUsedAnyNameLocation: new	192	\@GLSxtrp: new	47
\glsFindWidestUsedAnyNameSymbol: new	189	\@gls@preglossaryhook: added \glossxtrsetpopts	105
\glsFindWidestUsedAnyNameSymbolLocation: new	190	\@glsfirst@: added check for nohyperfirst attribute	26
\glsFindWidestUsedLevelTwo: new	187	\@glsfirstplural@: added check for nohyperfirst attribute	28
\glsFindWidestUsedTopLevelName: new	186	\@glsxtrinmark: new	160
\glsfirstlongfootnotefont: new	133	\@glsxtrnotinmark: new	160
\glsgetwidestname: new	185	\@glsxtrp: new	47
\glsgetwidestsubname: new	186	\@glsxtrp@opt: new	46
\glslongfootnotefont: new	133	\glossxtrsetpopts: new	46
\glsxtrAltTreeIndent: new	185	\glsps: new	49
\glsxtralttreeInit: new	185	\glspt: new	49
\glsxtrAltTreePar: new	185	\glsxtr@entry@p: new	48
\glsxtrAltTreeSetHangIndent: new	193	\glsxtrabbrvfootnote: new	133
\glsxtrAltTreeSetSubHangIndent: new	193	\glsxtrchecknohyperfirst: new	26
\glsxtralttreeSubSymbolDescLocation: new	185	\glsxtrfieldtitlecasecs: new	92
\glsxtralttreeSymbolDescLocation: new	184	\glsxtrifinmark: new	160
\glsxtrComputeTreeIndent: new	193	\GLSxtrp: new	50
\glsxtrComputeTreeSubIndent: new	193	\Glsxtrp: new	49
\glsxtrtreetopindent: new	185	\glsxtrp: new	48
short-em-long: fixed incorrect font used by long form	151	\glsxtrsetpopts: new	46
\xglsssetwidest: new	185	short-long-desc: added text key	132
1.06 (2016-06-18)		fixed misspelling of \glsabbrvfont in plural key	132
\@glsdoifexistsorwarn: new	6	long-short-desc: added missing text key	131
\@glsxtr@docdefval: new	5	fixed misspelling of \glsabbrvfont	131
\@glsxtr@usesee: new	14	footnote: changed first forms to use \glsfirstlongfootnotefont	133
General: disabled docdef key at the start of the document	12	postfootnote: removed \footnote from first keys	135
docdef option changed to choice	5	switched from \glsfirstlongfont to \glsfirstlongfootnotefont	136
\glsxtr@usesee: new	14	\RestoreAcronyms: modified	
\glsxtrusesee: new	14	\@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse	66
\glsxtruseseeformat: new	14		
\if@glsxtrdocdefrestricted: new	6		
1.07 (2016-08-15)			
\@glsxtrp: new	46		

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	
\.	106, 184
\@@cGLS@	54, 62
\@@cGLSpl@	54, 62
\@@cGLspl@	54, 62
\@@cglS@	54, 62
\@@cglSpl@	54, 55, 62
\@@glo@assign@sortkey	69
\@@glo@no@assign@sortkey	69
\@@glslocalreset	52
\@@glslocalunset	52
\@@glsreset	52
\@@glsunset	51
\@@glsxtr@autoindex@escspch	100–102
\@@glsxtr@checkspch	99, 100, 102
\@@glsxtr@disabledflycommand	20
\@@glsxtrp	47
\@@glsxtrpostloctag	22
\@@glsxtrpreloctag	22, 23
\@@newglossaryentry@defcounters	53
\@@newglossaryentry@defunitcounters	60
\@@par	185
\@ACRlong	43
\@ACRlongpl	43
\@ACRshort	43
\@ACRshortpl	43
\@Acrlong	43
\@Acrlongpl	43
\@Acrshort	43
\@Acrshortpl	43
\@GLS@	43, 56, 57
\@GLSdesc@	30
\@GLSpl@	43, 57
\@GLSplural@	44
\@GLSsymbol@	31
\@GLStext@	44
\@GLSxtr@full	115
\@GLSxtr@fullpl	117
\@GLSxtr@p@acrlong@	43
\@GLSxtr@p@acrlongpl@	43
\@GLSxtr@p@acrshort@	43
\@GLSxtr@p@acrshortpl@	43
\@GLSxtr@p@long@	43
\@GLSxtr@p@longpl@	43
\@GLSxtr@p@plural@	43
\@GLSxtr@p@short@	43
\@GLSxtr@p@shortpl@	43
\@GLSxtr@p@text@	43
\@GLSxtrlong	43, 120
\@GLSxtrlongpl	43, 123
\@GLSxtrp	50, 51
\@GLSxtrshort	43, 118
\@GLSxtrshortpl	43, 122
\@Gls@	43, 56, 57
\@Gls@acentryname	65
\@Gls@entry@field	37, 49, 50
\@Gls@entryname	65
\@GlsXtrEnableOnTheFly	17
\@Glspl@	43, 56, 57
\@Glsplural@	44
\@Glstext@	43
\@Glsxtr	18, 19
\@Glsxtr@full	115
\@Glsxtr@fullpl	116
\@Glsxtr@p@acrlong@	43
\@Glsxtr@p@acrlongpl@	43
\@Glsxtr@p@acrshort@	43
\@Glsxtr@p@acrshortpl@	43
\@Glsxtr@p@long@	43
\@Glsxtr@p@longpl@	43
\@Glsxtr@p@plural@	43
\@Glsxtr@p@short@	43
\@Glsxtr@p@shortpl@	43
\@Glsxtr@p@text@	43

\@Glsxtrlong	43, 119	\@gls@alt@hyp@opt	42
\@Glsxtrlongpl	43, 123	\@gls@alt@hyp@opt@char	41, 42
\@Glsxtrp	49, 50	\@gls@alt@hyp@opt@keys	42
\@Glsxtrpl	19	\@gls@automake	69
\@Glsxtrshort	43, 118	\@gls@checkedmkidx	99, 100, 102
\@Glsxtrshortpl	43, 121	\@gls@checkmkidxchars	99
\@acrlong	43	\@gls@codepage	72
\@acrlongpl	43	\@gls@declareoption	4
\@acrshort	43	\@gls@doautomake	69
\@acrshortpl	43	\@gls@encapchar	100
\@alt@gls@hyp@opt	41	\@gls@entry@count	54
\@auxout	23, 24, 54, 63, 67, 72	\@gls@entry@field	37, 48–51, 53
\@cGLS	57	\@gls@entry@unitcount	62, 63
\@cGLS@	54, 57, 62	\@gls@field@font	25–33
\@cGLSpl	57	\@gls@field@link	26–33, 38, 39
\@cGLSpl@	54, 57, 62	\@gls@hyp@opt	38, 39, 42, 57, 114–123
\@cGLspl@	54, 62	\@gls@hyp@opt@cs	41, 42
\@cgls@	54, 57, 62	\@gls@increment@currcount	53
\@cglspl@	54, 62	\@gls@increment@currunitcount	61
\@disable@onlypremakeg	68	\@gls@keymap	14, 37
\@do@auxoutstuff	71, 72	\@gls@label	41, 126
\@do@newglossaryentry	65, 112	\@gls@levelchar	100
\@empty	25, 33–37, 99, 100, 114–123	\@gls@link	25, 33–37, 114–124
\@end@glxstr@addunused	15	\@gls@link@checkfirsthyper	66
\@end@glxstr@usesee	14	\@gls@link@nocheckfirsthyper	25, 33–37, 114–123
\@endfortrue	126	\@gls@local@increment@currcount	53
\@firstofone	25, 26, 93, 94, 98, 104	\@gls@local@increment@currunitcount	61
\@firstofthree	25, 33–36, 41, 42, 114, 116, 117, 119, 121, 122	\@gls@loclist	70, 71
\@firstoftwo	26–31, 34–37, 39, 41, 67, 108, 109, 115–117, 121–123, 160, 161	\@gls@longpl	110–112
\@for	10, 15, 53, 64, 68, 69, 91, 103	\@gls@noidx@nosanitizesort	69
\@glo@assign@sortkey	69	\@gls@noidx@sanitizesort	69
\@glo@category	58	\@gls@noidx@loclist@finalsep	70
\@glo@countunit	58	\@gls@noidx@loclist@prev	70
\@glo@default@sorttype	69	\@gls@noidx@loclist@sep	70
\@glo@label	14, 15, 37, 186–193	\@gls@org@glsnoidxdisplayloc	70, 71
\@glo@name	99	\@gls@org@glsseeformat	70, 71
\@glo@parent	187–189	\@gls@preglossaryhook	103
\@glo@see	14, 15	\@gls@prevlevel	193, 194
\@glo@sort	99	\@gls@quotechar	99
\@glo@sorttype	69	\@gls@reference	16, 67
\@glo@thisvalue	156	\@gls@setdefault@glslink@opts	40
\@glo@tmp	37	\@gls@short	111
\@glo@type	15, 65, 68, 69, 71, 72, 75, 76	\@gls@shortpl	109, 111
\@glo@types	90, 186–192	\@gls@tmpb	102
\@glossary@default@style	20, 21, 195	\@gls@type	69, 126, 186–192
\@gls@	43, 55, 57	\@gls@write@entrycounts	54
\@gls@actualchar	99	\@gls@write@entryunitcounts	62
		\@gls@write@entryunitcounts@do	63

\@glsabbrv@current@abbreviation	110, 124	\@glsxtr@doaccsupp	9, 11
\@glsacronymlists	65	\@glsxtr@docdefval	5, 6, 16
\@glsdoifexistsorwarn	6, 94–97	\@glsxtr@doloctag	22, 23
\@glsentry	54, 63	\@glsxtr@dostylewarn	126
\@glslink	42, 43	\@glsxtr@enabletagging	103
\@glsnumberformat	98, 99	\@glsxtr@end@	17
\@glsorder	67	\@glsxtr@endescspch	99–102
\@glspl@	43, 56, 57	\@glsxtr@entrycount@org@localreset	54
\@glsplural@	44	\@glsxtr@entrycount@org@localunset	53
\@gls punc@token	108	\@glsxtr@entrycount@org@reset	54
\@glsstyle@alttree	184	\@glsxtr@entrycount@org@unset	53
\@glsstyle@inline	184	\@glsxtr@entryunitcount@org@localreset	62
\@glsstyle@listdotted	179	\@glsxtr@entryunitcount@org@localunset	61
\@glstarget	42	\@glsxtr@entryunitcount@org@reset	61
\@glstext@	43	\@glsxtr@entryunitcount@org@unset	61
\@glswidestname	185, 186, 193	\@glsxtr@field@linkdefs	25
\@glsxtr	18, 19	\@glsxtr@format@overridefalse	98
\@glsxtr@abbreviationsdef	7, 11	\@glsxtr@format@overridetrue	98
\@glsxtr@activate@initialtagging	103, 105	\@glsxtr@foundinlist	108
\@glsxtr@addunitcounter	58	\@glsxtr@full	114
\@glsxtr@addunusedxrefs	15	\@glsxtr@fullpl	116
\@glsxtr@attrval	92–97, 99	\@glsxtr@glossdescfont	92–94
\@glsxtr@autoindex@at	99, 100	\@glsxtr@glossnamefont	94–97
\@glsxtr@autoindex@doextra@esc	99	\@glsxtr@gobbleto@endescspch	102
\@glsxtr@autoindex@encap	99–101	\@glsxtr@idx@displaynumberlist	68
\@glsxtr@autoindex@esc	99, 101, 102	\@glsxtr@idx@entrynumberlist	68
\@glsxtr@autoindex@escat	100	\@glsxtr@ifcsstart	17
\@glsxtr@autoindex@escencap	100, 101	\@glsxtr@ifpunctoken	108
\@glsxtr@autoindex@esclevel	100, 101	\@glsxtr@ifunitcounter	58
\@glsxtr@autoindex@escquote	99, 101	\@glsxtr@insert@dots	110
\@glsxtr@autoindex@level	100, 101	\@glsxtr@insert@dots@next	110
\@glsxtr@autoindex@setname	99	\@glsxtr@insert@dots	111
\@glsxtr@autoindex@crossrefs	6, 14	\@glsxtr@label	15, 91, 92
\@glsxtr@cat	53, 64, 103	\@glsxtr@loadstyles	178
\@glsxtr@csname	59, 60, 62	\@glsxtr@noidx@displaynumberlist	68
\@glsxtr@current@style	20, 195	\@glsxtr@noidx@entrynumberlist	68
\@glsxtr@currentunitcount	59, 60, 62	\@glsxtr@noidx@numberlistloop	69
\@glsxtr@currunitcount	61, 63	\@glsxtr@notfoundinlist	108
\@glsxtr@declareoption	4, 7, 9	\@glsxtr@optlist	19
\@glsxtr@defaultnoglossarywarning	10	\@glsxtr@org@Glsxtrtitlefirst	161, 162
\@glsxtr@deprecated@abbrstyle	144, 145, 148, 149, 153–155	\@glsxtr@org@Glsxtrtitlefirstplural	161, 162
\@glsxtr@disabledflycommand	19	\@glsxtr@org@Glsxtrtitlefull	161, 162
\@glsxtr@do@@wrindex	41	\@glsxtr@org@Glsxtrtitlefullpl	161, 162
\@glsxtr@do@glsglisablehyperinlist	39, 40	\@glsxtr@org@Glsxtrtitlelong	161, 162
\@glsxtr@do@style	11, 177	\@glsxtr@org@Glsxtrtitlelongpl	161, 162
\@glsxtr@do@titlecaps@warn	93–96, 104	\@glsxtr@org@Glsxtrtitleplural	160, 162
\@glsxtr@doabbreviationsdef	7		

<code>\@glstr@org@Glsxtrtitleshort</code>	160, 161	<code>\@glstr@thisloctag</code>	23
<code>\@glstr@org@Glsxtrtitleshortpl</code>	160, 161	<code>\@glstr@tmp</code>	10
<code>\@glstr@org@Glsxtrtitletext</code>	160, 162	<code>\@glstr@type</code>	92
<code>\@glstr@org@MakeUppercase</code>	160, 161	<code>\@glstr@unitcountlist</code>	59
<code>\@glstr@org@checkfirsthyper</code>	39, 67	<code>\@glstr@usesee</code>	14
<code>\@glstr@org@delimN</code>	23	<code>\@glstrdocdeffalse</code>	16
<code>\@glstr@org@delimR</code>	23	<code>\@glstrindexcrossrefsfalse</code>	6
<code>\@glstr@org@glsgignore</code>	23	<code>\@glstrindexcrossrefstrue</code>	6
<code>\@glstr@org@Glsxtrtitlefirst</code>	161, 162	<code>\@glstrinmark</code>	159, 160
<code>\@glstr@org@Glsxtrtitlefirstplural</code>	161, 162	<code>\@glstrlong</code>	43, 119
<code>\@glstr@org@Glsxtrtitlefull</code>	161, 162	<code>\@glstrlongpl</code>	43, 122
<code>\@glstr@org@Glsxtrtitlefullpl</code>	161, 162	<code>\@glstrnotinmark</code>	159, 160
<code>\@glstr@org@Glsxtrtitlelong</code>	161, 162	<code>\@glstrtp</code>	48, 49
<code>\@glstr@org@Glsxtrtitlelongpl</code>	161, 162	<code>\@glstrtp@opt</code>	46
<code>\@glstr@org@Glsxtrtitleplural</code>	160, 162	<code>\@glstrtpl</code>	18, 19
<code>\@glstr@org@Glsxtrtitleshort</code>	160, 161	<code>\@glstrtpostloctag</code>	22, 24
<code>\@glstr@org@Glsxtrtitleshortpl</code>	160, 161	<code>\@glstrpreloctag</code>	22, 24
<code>\@glstr@org@Glsxtrtitletext</code>	160, 162	<code>\@glstrshort</code>	43, 117
<code>\@glstr@org@makeglossaries</code>	67	<code>\@glstrshortpl</code>	43, 120, 121
<code>\@glstr@org@markboth</code>	160	<code>\@glstrundeftag</code>	5, 12
<code>\@glstr@org@markright</code>	159, 160	<code>\@gobble</code>	7, 110
<code>\@glstr@org@newacronymstyle</code>	66	<code>\@gobbletwo</code>	109
<code>\@glstr@org@postdescription</code>	105	<code>\@ifnextchar</code>	41
<code>\@glstr@org@setacronymstyle</code>	66	<code>\@ifpackageloaded</code>	4, 7, 76, 92, 94, 96, 98, 176
<code>\@glstr@org@printglossary</code>	19	<code>\@ifstar</code>	17, 41, 103
<code>\@glstr@org@warndep</code>	109	<code>\@ifundefined</code>	176
<code>\@glstr@p@acrlong@</code>	43	<code>\@input@</code>	71
<code>\@glstr@p@acrlongpl@</code>	43	<code>\@istfilename</code>	67
<code>\@glstr@p@acrshort@</code>	43	<code>\@makeglossary</code>	68
<code>\@glstr@p@acrshortpl@</code>	43	<code>\@mfu@domakefirstuc</code>	104
<code>\@glstr@p@long@</code>	43	<code>\@mfu@nocaplist</code>	104
<code>\@glstr@p@longpl@</code>	43	<code>\@one</code>	54, 63
<code>\@glstr@p@plural@</code>	43	<code>\@newglossaryentry@defcounters</code>	53, 60
<code>\@glstr@p@short@</code>	43	<code>\@newglossaryentryposthook</code>	37
<code>\@glstr@p@shortpl@</code>	43	<code>\@newglossaryentryprehook</code>	37
<code>\@glstr@p@text@</code>	43	<code>\@nnil</code>	99, 100, 102, 108–110
<code>\@glstr@pagetag</code>	22, 23	<code>\@no@makeglossaries</code>	75
<code>\@glstr@pagetag</code>	22, 23	<code>\@onelevel@sanitize</code>	19
<code>\@glstr@prevunitcount</code>	61	<code>\@onlypreamble</code>	20, 23, 63, 98, 100, 101, 103
<code>\@glstr@redefstyles</code>	10, 177	<code>\@printgloss@setsort</code>	69
<code>\@glstr@reg@glosslist</code>	67–69	<code>\@printglossary</code>	19
<code>\@glstr@savepreloctag</code>	22–24	<code>\@sGlsXtrEnableOnTheFly</code>	17
<code>\@glstr@setentrycountunsetattr</code>	52	<code>\@secondofthree</code>	26–28, 33–36, 38, 115, 116, 118, 120, 121, 123
<code>\@glstr@setentryunitcountunsetattr</code>	64	<code>\@secondoftwo</code>	25, 29–37, 39, 42, 67, 109, 114, 115, 117–123, 136, 160, 161
<code>\@glstr@setupshortcuts</code>	8, 9, 11	<code>\@thirdofthree</code>	26–28, 34–37, 39, 115, 117, 118, 120, 122, 123
<code>\@glstr@swaptwo</code>	109	<code>\@thirdoftwo</code>	29–33
<code>\@glstr@tag</code>	103		
<code>\@glstr@taggingcs</code>	103		

\@warn@nomakeglossaries	72	\As	8
\@xdy@main@language	72	\as	7
\@xdy@language	72	\ASP	8
		\Asp	8
		\asp	7
_	73, 74	\AtBeginDocument	12, 21
		\AtEndDocument	15, 54, 62, 72
A			
\AB	8	B	
\Ab	7	babel package	99, 100, 108
\ab	7	\begin	57, 73, 74, 179–183, 193
abbreviation styles:		C	
long-noshort	153, 154	category attributes:	
short	138	discardperiod	107
\abbreviationsname	7	entrycount	51–54, 64
\abbrvpluralsuffix	111, 130, 132, 134, 136, 137, 139, 140, 142–149, 157, 158	firstuc	96
\ABP	8	glossdesc	92
\Abp	7	glossdescfont	92
\abp	7	glossname	94
\ACRfullfmt	65	glossnamefont	94, 97
\Acrfullfmt	65	headuc	162
\acrfullfmt	65	indexname	99
\ACRfullplfmt	65	indexonlyfirst	40
\Acrfullplfmt	65	insertdots	111
\acrfullplfmt	65	nohyper	39
\acronymentry	65	nohyperfirst	26–28
\acronymfont	33–37, 45, 46, 66	regular	24, 58, 129, 131, 133, 135, 138, 139, 141, 150, 152, 157, 158
\acronymname	7	\cGLS	8, 52, 64
\acronymsort	65	\cGls	7, 52, 64
\acronymtype	7, 65, 66	\cglS	7, 52, 64
\acrpluralsuffix	65	\cGLSformat	56
\actualchar	101	\cGlsformat	56
\addtolength	194	\cglSformat	55, 57
\advance	54, 63	\cGLSpl	8, 52, 64
\AF	8	\cGlspl	7, 52, 64
\Af	8	\cglSpl	7, 52, 64
\af	7	\cGLSplformat	56
\AFP	8	\cGlsplformat	56
\Afp	8	\cglSplformat	55, 58
\afp	7	\columnwidth	21
\AL	8	\count@	54, 55, 63
\Al	8	\cs	57, 193
\al	7	\csdef	37–39, 54, 59, 60, 62, 87, 126–128, 135, 179
\ALP	8	\csedef	60
\Alp	8	\csgdef	16, 22, 54, 59, 62
\alp	7	\csletcs	128
\AnyTrackedLanguages	176	\csname	20, 24, 33–39, 46, 60, 71, 72, 75, 76, 92, 109, 114–124, 128, 129, 193
\appto	10, 14, 37, 41, 53, 60, 98, 108, 110		
\AS	8		

<code>\csuse</code>	23, 38, 48–50, 58–62, 87, 98, 105, 106, 127, 128, 186–189
<code>\csxdef</code>	14, 59, 62
<code>\CurrentOption</code>	11, 178
<code>\CurrentTrackedTag</code>	177
<code>\CustomAbbreviationFields</code>	112, 129–133, 135, 137, 138, 140, 141, 150, 152, 153, 156, 158
D	
<code>\DeclareAcronymList</code>	65
<code>\DeclareOption</code>	4, 178
<code>\DeclareOptionX</code>	4, 11
<code>\def</code>	12, 14, 15, 17– 19, 22, 24–37, 42–46, 55–57, 65, 68–70, 98–104, 108–111, 114–123, 126, 193, 194
<code>\define@boolkey</code>	6, 40
<code>\define@choicekey</code>	5, 6, 8, 10
<code>\define@key</code>	10, 11, 37, 109, 110
<code>\DefineAcronymSynonyms</code>	9
<code>\delimN</code>	23
<code>\delimR</code>	23
<code>\detokenize</code>	17
<code>\dimen@</code>	67, 186–193
<code>\dimen@i</code>	187–189
<code>\dimen@ii</code>	187–189
<code>\dimexpr</code>	21, 185
<code>\disable@keys</code>	7, 12, 16
<code>\do</code>	10, 15, 53, 64, 68, 69, 91, 103
<code>\do@gls@link@checkfirsthyper</code>	25, 33–37, 114–123
<code>\do@gls@disablehyperinlist</code>	40
<code>doc</code> package	101
<code>\DTLifinlist</code>	68, 69
E	
<code>\eappto</code>	10, 99, 178
<code>\edef</code>	39, 58–60, 62, 67, 69, 71, 72, 92–95, 97, 99, 100, 102, 109, 187–189
<code>\eglssetwidest</code>	186–193
<code>\else</code>	6, 7, 9, 10, 16, 17, 22, 24, 40, 55, 67, 69, 73, 75, 98–100, 102, 108, 110, 118–123, 130, 132, 134–141, 157–159, 180–184, 194, 195
<code>\emph</code>	149
<code>\encapchar</code>	102
<code>\end</code>	73, 74, 179–183, 193
<code>\endcsname</code>	20, 24, 33–39, 46, 60, 71, 72, 75, 76, 92, 109, 114–124, 128, 129, 193
entry categories:	
abbreviation	124
general	86, 88
index	90
<code>\epreto</code>	99
<code>\equal</code>	75
<code>etoolbox</code> package	4
<code>\expandafter</code>	11, 14, 15, 17–19, 37–39, 41, 42, 46, 57, 58, 68, 69, 92, 95, 96, 99, 101, 102, 108, 111, 112
<code>\expandonce</code>	65, 99, 100
F	
<code>\fi</code>	5–7, 9, 10, 15–17, 20–22, 24, 40, 41, 55, 62, 63, 67, 69, 70, 72, 73, 75, 98–100, 102, 108, 110, 117–123, 130, 132, 134–141, 157–159, 180–184, 186–195
<code>\firstacronymfont</code>	66, 67
<code>\footnote</code>	133
<code>\foralllglossaries</code>	15, 90, 92, 186–192
<code>\foralllglsentries</code>	54, 63
<code>\ForEachTrackedDialect</code>	177
<code>\forlglsentries</code>	15, 90, 92, 186–192
<code>\forlistcsloop</code>	63
<code>\forlistloop</code>	70, 71, 104
<code>\futurelet</code>	108
G	
<code>\gdef</code>	23, 100, 101
<code>\Genacrfullformat</code>	65
<code>\genacrfullformat</code>	65
<code>\GenericAcronymFields</code>	65
<code>\Genplacrfullformat</code>	65, 66
<code>\genplacrfullformat</code>	65
<code>\glo@name</code>	95, 96
<code>glossaries</code> package	178
<code>glossaries-accsupp</code> package	9, 11, 76
<code>glossaries-extra</code> package	2
<code>glossaries-extra-stylemods</code> package	10, 105, 177
<code>\GlossariesExtraWarning</code>	5, 7, 17, 19, 66, 73, 93, 94, 96, 97, 103, 104, 128
<code>\GlossariesExtraWarningNoLine</code>	7, 55, 63
<code>\GlossariesWarning</code>	22, 70, 71, 126
<code>\GlossariesWarningNoLine</code>	68, 72
glossary styles:	
alttree	184, 185, 193
inline	184
listdotted	179
listdottedstyle	179
sublistdotted	179

glossary-long package	180	\glsaccesslongpl	36, 37, 113, 122,
glossary-longbooktabs package	180		123, 130, 132, 134–137, 139–141, 157–159
glossary-mcols package	178	\GLSaccessname	29
glossary-tree package	178	\Glsaccessname	29
\glossaryentrynumbers	24	\glsaccessname	28
\glossaryheader	179–183, 194	\GLSaccessplural	27
\glossarysection	75	\Glsaccessplural	27
\glossarytitle	75	\glsaccessplural	27
\glossarytoctitle	75	\Glsaccessshort	
\glossentry	179–183, 194		33, 118, 125, 132, 134–136, 139, 158
\glossentrydesc	179–185	\glsaccessshort ...	33, 34, 112, 117, 119,
\glossentryname	179–183, 194		124, 125, 130, 132, 134, 136–140, 157, 158
\glossentrysymbol	180–185	\Glsaccessshortpl	
\glossxtrsetpopts	105		34, 121, 124, 132, 134–137, 139, 158
\GLS	52, 64	\glsaccessshortpl	34, 35, 113, 121,
\Gls	18, 52, 64		122, 124, 130, 132, 134, 136–141, 157, 158
\gls	18, 20, 52, 64, 73	\GLSaccesssymbol	30
\gls@assign@field	37	\Glsaccesssymbol	30, 103
\gls@checkseeallowed	16, 17, 68	\glsaccesssymbol	30, 103, 107
\gls@codepage	72	\GLSaccesssymbolplural	31
\gls@defdocnewglossaryentry	53, 61	\Glsaccesssymbolplural	31
\gls@defglossaryentry	18, 19	\glsaccesssymbolplural	30
\gls@save@numberlist	22, 24	\GLSaccessstext	26
\gls@tmplen	186–192, 194	\Glsaccessstext	26
\glsabbrvdefaultfont		\glsaccessstext	26
	114, 130, 132, 134, 136, 137, 139, 140	\glsacrshortcutstrue	9
\glsabbrvmfont	149–155	\glsacspacemax	67
\glsabbrvfont	44, 45, 66,	\glsadd	15, 73
	114, 117–119, 121, 122, 124, 125, 129–158	\glsaddstoragekey	86
\glsabbrvuserfont	156–158	\glsbackslash	17
\glsabrvfont		\glsapscase	25–39, 114–125
	129, 131, 133, 135, 150, 152, 156, 158	\glsacategory	24, 25, 39, 44, 45, 87–89, 92–
\GLSaccessdesc	29		98, 102, 103, 105, 106, 114–118, 121, 122
\Glsaccessdesc	29, 93, 102	\glsacategorylabel	39, 109–111, 135
\glsaccessdesc	29, 93, 106	\glsaclosebrace	74, 75
\GLSaccessdescplural	30	\glsacurrententrylabel	22, 23, 98, 104, 105
\Glsaccessdescplural	30	\glsacurrentfieldvalue	155, 156
\glsaccessdescplural	29	\glsacustomtext	25, 33–37, 114–124, 126
\GLSaccessfirst	27	\glsadefaulttype	7, 73
\Glsaccessfirst	27	\glsadescriptionaccessdisplay	80, 93
\glsaccessfirst	26	\glsadescriptionpluralaccessdisplay	81
\GLSaccessfirstplural	28	\glsadescwidth	179, 181–183
\Glsaccessfirstplural	28	\glsadetoklabel	12–17,
\glsaccessfirstplural	28		53, 54, 59–63, 70, 71, 92, 95, 96, 187–189
\Glsaccesslong	35, 112, 120, 130, 138, 140, 157	\glsdisplaynumberlist	68, 70
\glsaccesslong	35, 36, 112, 119,	\glsdohyperlink	43
	120, 130, 132, 134–137, 139–141, 157, 158	\glsdoifexists	6, 14, 25, 33–37, 70, 71, 114–123
\Glsaccesslongpl		\glsdoifexistsorwarn	6, 92, 93, 102, 103
	36, 113, 123, 130, 138, 141, 157	\glsdonohyperlink	42, 43

<code>\glsdosanitizesort</code>	69	<code>\glsentryuserv</code>	32
<code>\glsenableentrycount</code>	52, 55, 63	<code>\Glsentryuservi</code>	33
<code>\glsenableentryunitcount</code>	54, 63, 64	<code>\glsentryuservi</code>	33
<code>\glsentrycurrcount</code>	53, 54, 61	<code>\glsfieldxdef</code>	92
<code>\Glsentrydesc</code>	80, 85, 94	<code>\glsfindwidesttoplevelname</code>	186
<code>\glsentrydesc</code>	80, 85, 94	<code>\GLSfirst</code>	165, 166
<code>\Glsentrydescplural</code>	81, 85	<code>\Glsfirst</code>	166
<code>\glsentrydescplural</code>	81, 85	<code>\glsfirst</code>	165, 166
<code>\Glsentryfirst</code>	58, 78, 84	<code>\glsfirstabbrvdefaultfont</code>	
<code>\glsentryfirst</code>	58, 78, 84, 173	113, 130, 132, 134, 136, 137, 139, 140
<code>\Glsentryfirstplural</code>	58, 79, 84	<code>\glsfirstabbrvemfont</code>	150–155
<code>\glsentryfirstplural</code>	58, 78, 79, 84, 173, 174	<code>\glsfirstabbrvfont</code> ..	66, 112, 113, 129–158
<code>\Glsentryfull</code>	65	<code>\glsfirstabbrvuserfont</code>	157, 158
<code>\glsentryfull</code>	65	<code>\glsfirstaccessdisplay</code>	78
<code>\Glsentryfullpl</code>	66	<code>\glsfirstlongdefaultfont</code>	
<code>\glsentryfullpl</code>	65	130, 132, 137, 139, 140
<code>\glsentryitem</code>	179–183, 194	<code>\glsfirstlongemfont</code>	150–152, 154, 155
<code>\Glsentrylong</code>	45, 46, 58, 82, 86	<code>\glsfirstlongfont</code>	112, 113,
<code>\glsentrylong</code> .	45, 46, 58, 82, 83, 86, 135, 174	129–132, 134, 136–141, 150–152, 154–159
<code>\Glsentrylongpl</code>	45, 46, 58, 83, 86	<code>\glsfirstlongfootnotefont</code>	133–137
<code>\glsentrylongpl</code> ..	45, 46, 58, 83, 86, 174, 175	<code>\glsfirstlonguserfont</code>	157, 158
<code>\Glsentryname</code>	76, 83, 94, 96, 97	<code>\GLSfirstplural</code>	166, 167
<code>\glsentryname</code>	76, 83, 99, 186–193	<code>\Glsfirstplural</code>	167
<code>\glsentrynumberlist</code>	68, 71, 191–193	<code>\glsfirstplural</code>	166
<code>\Glsentryplural</code>	77, 84	<code>\glsfirstpluralaccessdisplay</code>	78, 79
<code>\glsentryplural</code>	77, 78, 84, 172	<code>\glsforeachincategory</code>	126
<code>\glsentryprevcount</code>	53, 55, 61	<code>\glsgenentryfmt</code>	24
<code>\glsentryprevmaxcount</code>	61	<code>\glsgetattribute</code> ...	55, 59–61, 92–95, 97, 99
<code>\glsentryprevtotalcount</code>	61	<code>\glsgetcategoryattribute</code>	88
<code>\Glsentryshort</code>	44, 45, 81, 85	<code>\glsgetwidestname</code>	185
<code>\glsentryshort</code>		<code>\glsgroupheading</code>	179–183, 194
.....	44, 45, 67, 81, 82, 85, 86, 170, 171	<code>\glsgroupskip</code>	180–184, 195
<code>\Glsentryshortpl</code>	45, 82, 86	<code>\glshasattribute</code>	
<code>\glsentryshortpl</code>	44–46, 82, 86, 171	54, 55, 59, 60, 62, 63, 92–95,
<code>\Glsentrysymbol</code>	79, 84	97, 99, 129, 131, 133–135, 150, 152, 157, 158
<code>\glsentrysymbol</code>	79, 84, 85, 190, 191	<code>\glshascategoryattribute</code>	88
<code>\Glsentrysymbolplural</code>	80, 85	<code>\glshypernumber</code>	98
<code>\glsentrysymbolplural</code>	79, 80, 85	<code>\glsifattribute</code>	
<code>\Glsentrytext</code>	77, 84	..	26, 40, 48, 90, 93–96, 104, 107, 162–170
<code>\glsentrytext</code>	77, 83, 84, 172	<code>\glsifcategory</code>	90
<code>\Glsentryuseri</code>	31	<code>\glsifcategoryattribute</code> ...	39, 88, 89, 111
<code>\glsentryuseri</code>	31	<code>\glsifnotregular</code>	26
<code>\Glsentryuserii</code>	31	<code>\glsifnotregularcategory</code>	89
<code>\glsentryuserii</code>	31	<code>\glsifplural</code> .	25, 27–31, 33–37, 107, 114–125
<code>\Glsentryuseriii</code>	32	<code>\glsifregular</code>	24, 25, 58
<code>\glsentryuseriii</code>	32	<code>\glsifregularcategory</code>	89
<code>\Glsentryuseriv</code>	32	<code>\glsignore</code>	23
<code>\glsentryuseriv</code>	32	<code>\glsinlinedescformat</code>	184
<code>\Glsentryuserv</code>	32	<code>\glsinlinesubdescformat</code>	184

<code>\glsinsert</code>	25, 33–37, 114–126	<code>\glspluralsuffix</code>	111, 114, 130, 132, 134, 136, 137, 139, 140, 142, 146, 156
<code>\glskeylisttok</code>	65, 110, 112	<code>\glspostdescription</code>	105, 179–185
<code>\glslabel</code> 24, 39, 40, 66, 106, 107, 124–126, 135		<code>\glspostinline</code>	184
<code>\glslabeltok</code>	65, 110, 112, 129–135, 137–141, 150, 152, 154, 156–158	<code>\glspostlinkhook</code>	25, 33–37, 46, 115–124
<code>\glsletentryfield</code>	99	<code>\glsprestandardsort</code>	69
<code>\glslink</code>	65	<code>\glsseeformat</code>	14, 70, 71
<code>\glslink options</code>		<code>\glssetabbrvfmt</code>	24, 25, 44, 45, 92–97, 102, 103, 114–118, 121, 122
format	98	<code>\glssetattribute</code>	129, 131, 133– 135, 137, 139–141, 150, 152, 154, 157, 158
hyper	159	<code>\glssetcategoryattribute</code>	53, 64, 67, 87–91, 103
noindex	40, 159	<code>\glsshortaccessdisplay</code>	81, 82
<code>\glslinkcheckfirsthyperhook</code>	39	<code>\glsshortpltok</code>	111, 112, 129, 131–133, 135, 137–139, 150, 152, 156, 158
<code>\glslinkvar</code>	41, 42	<code>\glsshortpluralaccessdisplay</code>	82
<code>\glslistdottedwidth</code>	179	<code>\glsshorttok</code>	65, 110–112, 129–133, 135, 137, 138, 141, 150, 152, 154, 156, 158
<code>\glslongaccessdisplay</code>	82, 83	<code>\glssubentryitem</code>	179–184, 194
<code>\glslongdefaultfont</code>	114, 130, 132, 133, 137, 139, 140	<code>\glssymbolaccessdisplay</code>	79
<code>\glslongemfont</code>	149–152, 154, 155	<code>\glssymbolpluralaccessdisplay</code> ...	79, 80
<code>\glslongfont</code>	45, 114, 119, 120, 122, 123, 130, 132, 134, 136, 137, 139, 140, 151, 152, 154, 155, 157, 158	<code>\glstarget</code>	179–184, 194
<code>\glslongfootnotefont</code>	133, 134, 136	<code>\GLStext</code>	164
<code>\glslongpltok</code>	112, 129, 131–133, 140, 141, 150, 152, 154, 156, 158	<code>\Glstext</code>	164
<code>\glslongpluralaccessdisplay</code>	83	<code>\glstext</code>	164
<code>\glslongtok</code>	65, 110, 112, 129–133, 135, 137, 139–141, 150, 152, 154, 156, 158	<code>\glstextaccessdisplay</code>	77
<code>\glslonguserfont</code>	156–158	<code>\glstextup</code>	142
<code>\glsnameaccessdisplay</code>	76, 94, 95, 97	<code>\glstreeindent</code>	193, 194
<code>\glsnamefont</code>	94, 96, 97	<code>\glstreenamebox</code>	194
<code>\glsnoidxdisplayloc</code>	70, 71	<code>\glstreenamefmt</code>	185–194
<code>\glsnoidxdisplayloclisthandler</code>	70	<code>\glstype</code>	33–37, 114–124
<code>\glsnoidxloclist</code>	71	<code>\glsunset</code>	15, 55, 56
<code>\glsnoidxnumberlistloophandler</code>	71	<code>\glswrite</code>	67
<code>\glsnonumberlistfalse</code>	22	<code>\Glsxtr</code>	19
<code>\glsnonumberlisttrue</code>	22	<code>\glxtr</code>	19
<code>\glsnumberlistloop</code>	68	<code>\glxtr@addunused</code>	15
<code>\glsnumlistlastsep</code>	70	<code>\glxtr@applyabbrvfmt</code>	124
<code>\glsnumlistsep</code>	70	<code>\glxtr@applyabbrvstyle</code>	109, 110, 126
<code>\glsopenbrace</code>	74, 75	<code>\glxtr@doption</code>	4, 7, 11
<code>\glsorder</code>	67	<code>\glxtr@headentry@p</code>	48, 49
<code>\glspagelistwidth</code>	179, 181–183	<code>\glxtr@ifnextpunc</code>	108
<code>\GLSpl</code>	52, 64	<code>\glxtr@ifpunctoken</code>	108
<code>\Glspl</code>	19, 52, 64	<code>\glxtr@keylist</code>	18, 19
<code>\glspl</code>	19, 52, 64	<code>\glxtr@next</code>	108, 109
<code>\GLSplural</code>	165	<code>\glxtr@orgmakenoidxglossaries</code>	16
<code>\Glsplural</code>	165	<code>\glxtr@punclist</code>	108
<code>\glsplural</code>	165	<code>\glxtr@usesee</code>	14
<code>\glspluralaccessdisplay</code>	77	<code>\glxtr@warnonexistsordo</code>	5, 13

<code>\glxtrabbrvfootnote</code>	133–135	<code>\Glsxtrheadfirstplural</code>	161
<code>\glxtrabbrvtype</code>	7, 112	<code>\glxtrheadfirstplural</code>	161
<code>\glxtraddallcrossrefs</code>	15	<code>\Glsxtrheadfull</code>	161
<code>\glxtrAltTreeIndent</code>	185	<code>\glxtrheadfull</code>	161
<code>\glxtralttreeInit</code>	193	<code>\Glsxtrheadfullpl</code>	161
<code>\glxtrAltTreePar</code>	184	<code>\glxtrheadfullpl</code>	161
<code>\glxtrAltTreeSetHangIndent</code> ...	185, 194	<code>\Glsxtrheadlong</code>	161
<code>\glxtrAltTreeSetSubHangIndent</code> ...	194	<code>\glxtrheadlong</code>	161
<code>\glxtralttreeSubSymbolDescLocation</code>	194	<code>\Glsxtrheadlongpl</code>	161
<code>\glxtralttreeSymbolDescLocation</code> ..		<code>\glxtrheadlongpl</code>	161
.....	185, 194	<code>\Glsxtrheadplural</code>	161
<code>\glxtrassignfieldfont</code>	26–33	<code>\glxtrheadplural</code>	161
<code>\glxtrcat</code>	18, 19	<code>\Glsxtrheadshort</code>	161
<code>\glxtrchecknohyperfirst</code>	26–28	<code>\glxtrheadshort</code>	161
<code>\glxtrComputeTreeIndent</code>	194	<code>\Glsxtrheadshortpl</code>	161
<code>\glxtrComputeTreeSubIndent</code>	194	<code>\glxtrheadshortpl</code>	161
<code>\GlsXtrDefineAbbreviationShortcuts</code> ..	9	<code>\Glsxtrheadtext</code>	161
<code>\GlsXtrDefineOtherShortcuts</code>	9	<code>\glxtrheadtext</code>	161
<code>\glxtrdiscardperiod</code>	106	<code>\glxtrtrifcounttrigger</code>	55, 56
<code>\glxtrdoautoindexname</code>	40, 41, 98	<code>\glxtrtrifemptyglossary</code>	75
<code>\glxtrdopostpunc</code>	135	<code>\glxtrtrifindexing</code>	40
<code>\glxtrdowrglossaryhook</code>	41	<code>\glxtrtrifinmark</code>	48–51, 160, 161
<code>\GlsXtrEnableEntryCounting</code>	64	<code>\glxtrtrifnextpunc</code>	108, 109
<code>\GlsXtrEnableEntryUnitCounting</code>	52	<code>\glxtrtrifperiod</code>	107
<code>\GlsXtrEnableOnTheFly</code>	17, 20	<code>\glxtrtrifwasfirstuse</code>	25–28, 33–
<code>\glxtrfieldtitlecase</code>	93–96	37, 39, 67, 106, 107, 115, 117–123, 135, 136	
<code>\glxtrfieldtitlecasecs</code>	92	<code>\Glsxtrinlinefullformat</code>	
<code>\glxtrfirstscfont</code>	142–145	113, 115, 127, 128, 135, 136, 138–140, 175	
<code>\glxtrfirstsmfont</code>	146–149	<code>\glxtrinlinefullformat</code>	
<code>\GlsXtrFormatLocationList</code> 22, 24, 191–193	 113–115, 127, 128, 134, 136–140, 175	
<code>\GLSxtrfull</code>	8, 169	<code>\Glsxtrinlinefullplformat</code>	
<code>\Glsxtrfull</code>	8, 170 113, 116, 127, 128, 135, 137–140, 176	
<code>\glxtrfull</code>	7, 169	<code>\glxtrinlinefullplformat</code> ..	113, 116,
<code>\Glsxtrfullformat</code> ... 113, 125, 127, 128,		117, 127, 128, 134, 136, 137, 139, 140, 176	
130, 132, 134, 136, 138, 139, 141, 157, 158		<code>\glxtrinsertinsidefalse</code>	129
<code>\glxtrfullformat</code>	113, 125–128,	<code>\GLSxtrlong</code>	8, 167, 168
130, 132, 134, 136, 138, 139, 141, 157, 158		<code>\Glsxtrlong</code>	8, 168
<code>\GLSxtrfullpl</code>	8, 169, 170	<code>\glxtrlong</code>	7, 167
<code>\Glsxtrfullpl</code>	8, 170	<code>\GLSxtrlongpl</code>	8, 167, 168
<code>\glxtrfullpl</code>	7, 169	<code>\Glsxtrlongpl</code>	8, 168
<code>\Glsxtrfullplformat</code> . 113, 125, 127, 128,		<code>\glxtrlongpl</code>	7, 167, 168
130, 132, 134, 136, 138, 139, 141, 157, 158		<code>\glxtrlongshortdescsort</code>	130
<code>\glxtrfullplformat</code> 125, 127, 128,		<code>\glxtrmarkhook</code>	159, 160
130, 132, 134, 136, 138, 139, 141, 157, 158		<code>\glxtrnewabbrevpresetkeyhook</code>	111
<code>\glxtrfullsep</code>		<code>\glxtrnewnumber</code>	8
112, 113, 129–132, 134–141, 150, 152, 156		<code>\glxtrnewsymbol</code>	8
<code>\glxtrgenabbrvfmt</code>	24	<code>\glxtrNoGlossaryWarning</code>	10, 71
<code>\Glsxtrheadfirst</code>	161	<code>\GlsXtrNoGlsWarningAutoMake</code>	75
<code>\glxtrheadfirst</code>	161	<code>\GlsXtrNoGlsWarningBuildInfo</code>	75

<code>\GlsXtrNoGlsWarningCheckFile</code>	75	<code>\glstxtrtitlefirstplural</code>	161, 162, 173
<code>\GlsXtrNoGlsWarningEmptyMain</code>	75	<code>\Glsxtrtitlefull</code>	161, 162, 175
<code>\GlsXtrNoGlsWarningEmptyNotMain</code>	75	<code>\glstxtrtitlefull</code>	161, 162, 175
<code>\GlsXtrNoGlsWarningEmptyStart</code>	75	<code>\Glsxtrtitlefullpl</code>	161, 162, 176
<code>\GlsXtrNoGlsWarningHead</code>	75	<code>\glstxtrtitlefullpl</code>	161, 162, 176
<code>\GlsXtrNoGlsWarningMisMatch</code>	75	<code>\Glsxtrtitlelong</code>	161, 162, 174
<code>\GlsXtrNoGlsWarningNoOut</code>	75	<code>\glstxtrtitlelong</code>	161, 162, 174
<code>\GlsXtrNoGlsWarningTail</code>	76	<code>\Glsxtrtitlelongpl</code>	161, 162, 175
<code>\glstxrorg@ifKV@glslink@hyper</code>	25	<code>\glstxtrtitlelongpl</code>	161, 162, 174, 175
<code>\GLSxtrp</code>	48	<code>\Glsxtrtitleplural</code>	160–162, 172, 173
<code>\Glsxtrp</code>	47	<code>\glstxtrtitleplural</code>	160–162, 172
<code>\glstxtrp</code>	47, 49	<code>\Glsxtrtitleshort</code>	160, 161, 171
<code>\Glsxtrpl</code>	19	<code>\glstxtrtitleshort</code>	160, 161, 170
<code>\glstxtrpl</code>	19	<code>\Glsxtrtitleshortpl</code>	160, 161, 171
<code>\glstxrpostdescription</code>	90, 105, 184	<code>\glstxtrtitleshortpl</code>	160, 161, 171
<code>\glstxrpostlink</code>	106	<code>\Glsxtrtitletext</code>	160–162, 172
<code>\glstxrpostlinkendsentence</code>	106	<code>\glstxtrtitletext</code>	160–162, 172
<code>\glstxrpostlinkhook</code>	106	<code>\glstxtrtreetopindent</code>	185, 193
<code>\glstxrpostlocalreset</code>	52, 54, 62	<code>\glstxrundefaction</code>	5, 12–14
<code>\glstxrpostlocalunset</code>	52, 53, 61	<code>\glstxrundeftag</code>	12
<code>\glstxrpostnamehook</code>	95–98	<code>\GlsXtrUseAbbrStyleFmts</code>	
<code>\GlsXtrPostNewAbbreviation</code>	131, 133, 141–155, 157, 159
.....	112, 127, 129, 131, 133,	<code>\GlsXtrUseAbbrStyleSetup</code>	142–155, 157, 159
135, 137, 139–141, 150, 152, 154, 157, 158		<code>\glstxruserfield</code>	156
<code>\glstxrpostreset</code>	52, 54, 61, 62	<code>\glstxruserparen</code>	156–159
<code>\glstxrpostunset</code>	51, 53, 61	<code>\glstxrusersuffix</code>	157, 158
<code>\glstxrprotectlinks</code>	42	<code>\glstxruseseeformat</code>	14
<code>\glstxrregularfont</code>	24–26	<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	109, 128
<code>\glstxrrestoremarkhook</code>	159, 160	<code>\GlsXtrWarning</code>	18, 19
<code>\glstxrscfont</code>	142–145		
<code>\glstxrscsuffix</code>	142–145		
<code>\GlsXtrSetActualChar</code>	101		
<code>\GlsXtrSetEncapChar</code>	102		
<code>\GlsXtrSetEscChar</code>	101		
<code>\GlsXtrSetLevelChar</code>	102		
<code>\glstxrsetpopts</code>	46		
<code>\glstxrsetupfulldefs</code>	114–117, 136		
<code>\GLSxtrshort</code>	8, 50, 51, 162, 163		
<code>\Glsxtrshort</code>	8, 163		
<code>\glstxtrshort</code>	7, 162		
<code>\GLSxtrshortpl</code>	8, 162, 163		
<code>\Glsxtrshortpl</code>	8, 163, 164		
<code>\glstxtrshortpl</code>	7, 163		
<code>\glstxrsmfont</code>	146–149		
<code>\glstxrsmsuffix</code>	146–149		
<code>\glstxrtagfont</code>	105		
<code>\Glsxtrtitlefirst</code>	161, 162, 173		
<code>\glstxtrtitlefirst</code>	161, 162, 173		
<code>\Glsxtrtitlefirstplural</code>	161, 162, 174		

H

<code>\hangindent</code>	185, 193
<code>\hbox</code>	179
<code>\hfill</code>	179
<code>\hsize</code>	21
<code>\hss</code>	179
<code>\hyperlink</code>	42
<code>\hyperpage</code>	98
<code>hyperref package</code>	42, 98, 159, 170

I

<code>\if</code>	17
<code>\if@glstxr@format@override</code>	99
<code>\if@glstxrdocdefrestricted</code>	16
<code>\if@glstxrindexcrossrefs</code>	6, 15
<code>\ifblank</code>	10, 18, 19, 67
<code>\ifcase</code>	5, 8, 10, 16
<code>\ifcsdef</code>	37, 38, 47–51,
59, 92–95, 97, 106, 109, 124, 127, 179–183	
<code>\ifcsstring</code>	12, 88, 126

<code>\ifcsundef</code>	16, 20, 22, 42, 53, 59–62, 71, 88, 126–129, 178, 186, 193
<code>\ifcsvoid</code>	87
<code>\ifdef</code> .. 8, 13, 20, 21, 39, 40, 48–50, 70, 71, 90, 91, 101, 102, 105, 155, 170–176, 179, 184	
<code>\ifdefempty</code> ...	14, 53, 64, 65, 68, 69, 103, 124
<code>\ifdefequal</code>	75
<code>\ifdefstring</code>	99, 104
<code>\ifdefvoid</code>	14, 15, 58
<code>\ifdim</code>	21, 67, 186–193
<code>\IfFileExists</code>	10, 71, 75, 178
<code>\ifglossaryexists</code>	12–14
<code>\ifglslacronym</code>	7, 75
<code>\ifglssautomake</code>	69, 75
<code>\ifglslentryexists</code>	12, 13, 18, 19, 22, 88, 105, 106
<code>\ifglslfieldeq</code>	86
<code>\ifglslhasfield</code>	156
<code>\ifglslhaslong</code>	58
<code>\ifglslhasparent</code>	186–189
<code>\ifglslhasshort</code>	24, 25
<code>\ifglslhassymbol</code>	107, 185
<code>\ifglslindexonlyfirst</code>	40
<code>\ifglslnogroupskip</code>	180–184, 195
<code>\ifglslnonumberlist</code>	24
<code>\ifglssanitizesort</code>	69
<code>\ifglslused</code>	15, 39, 40, 54, 63, 66, 124, 186, 187, 189, 191, 192
<code>\ifglslxindy</code>	71, 73
<code>\ifglslxtrinsertinside</code>	117–123, 130, 132, 134–141, 157–159
<code>\ifHy@hyperindex</code>	98
<code>\ifinlistcs</code>	16
<code>\ifKV@glslink@hyper</code>	25
<code>\ifKV@glslink@noindex</code>	40
<code>\ifnum</code>	5, 6, 55, 62, 63, 194
<code>\ifthenelse</code>	75
<code>\IfTrackedLanguageFileExists</code>	177
<code>\ifundef</code>	42, 67, 103, 104
<code>\ifx</code>	20, 22, 99, 100, 102, 108, 110, 195
<code>\immediate</code>	54, 63, 72
<code>\index</code>	99
<code>\indexspace</code>	195
<code>\input</code>	176
<code>\istfilename</code>	67
<code>\item</code>	73, 74, 179
J	
<code>\jobname</code>	71, 73–76
K	
<code>\key@ifundefined</code>	37
<code>\KV@glslink@hyperfalse</code> ..	26, 39, 40, 42, 43
<code>\KV@glslink@noindexfalse</code>	40
<code>\KV@glslink@noindextrue</code>	43
L	
<code>\LaTeX</code>	73, 74
<code>\leaders</code>	179
<code>\let</code>	4, 7, 8, 11, 16, 17, 19–23, 25–39, 41–43, 46, 52–54, 61–68, 70, 71, 93, 94, 96–100, 103–105, 108–111, 114–123, 136, 159–162, 184, 186
<code>\letabbreviationstyle</code>	135, 137, 138, 140–144, 147, 148, 153
<code>\letcs</code>	14, 15, 37, 70, 71, 92–97
<code>\levelchar</code>	102
<code>\listadd</code>	59
<code>\listbreak</code>	104
<code>\listcseadd</code>	60
<code>\listcsgadd</code>	16
<code>\listcsxadd</code>	59
<code>\loadglslentries</code>	16, 73
M	
<code>\MakeAcronymsAbbreviations</code>	66
<code>\makeatletter</code>	71, 101
<code>\makeatother</code>	100
<code>\makebox</code>	179, 194
<code>\makefirstuc</code>	104
<code>\makeglossaries</code>	67, 72, 74, 75
<code>\makeglossary</code>	68
<code>\makeindex</code>	67
<code>\makenoidxglossaries</code>	74
<code>\MakeTextUppercase</code>	161
<code>\MakeUppercase</code>	160, 161
<code>\marg</code>	193
<code>\markboth</code>	160
<code>\markright</code>	160
<code>\maxdimen</code>	21
<code>\mbox</code>	194
<code>\medskip</code>	75
<code>\MessageBreak</code>	16, 20, 55, 63, 69, 126
<code>mfirstuc package</code>	104
<code>\mfirstucMakeUppercase</code>	26– 37, 39, 44–46, 48, 50, 51, 57, 58, 65, 76–86, 95, 96, 115, 117, 119, 120, 122–126
<code>\mfu@checkword@arg</code>	104
<code>\mfu@checkword@do</code>	104

N	
<code>\NeedsTeXFormat</code>	4, 178
<code>\new@glossaryentry</code>	16, 69
<code>\new@ifnextchar</code>	38, 57, 108, 114–123
<code>\newabbr</code>	8
<code>\newabbreviation</code>	8, 66
<code>\newabbreviationhook</code>	112
<code>\newabbreviationstyle</code>	129–133, 135, 137, 138, 140–159
<code>\newacronym</code>	65, 66
<code>\newacronymhook</code>	65
<code>\newacronymstyle</code>	66
<code>\newcommand</code>	4–15, 17–20, 22–26, 37, 38, 40–43, 46, 48–53, 55, 57–61, 63, 64, 66, 67, 70–92, 98–110, 112–124, 126–130, 133, 142, 146, 149, 155, 156, 160–176, 178, 184–186, 193
<code>\newcount</code>	5
<code>\newentry</code>	8
<code>\newglossary</code>	7, 68
<code>\newglossaryentry</code>	8, 16, 53, 61, 65, 90, 91, 112
<code>\newglossaryentry</code> options	
<code>desc</code>	80, 85
<code>descplural</code>	81, 85
<code>first</code>	42, 78, 84, 129, 165–167, 173
<code>firstplural</code>	78, 79, 84, 129, 166, 173
<code>long</code>	82, 86, 174
<code>longplural</code>	83, 86, 174
<code>name</code>	76, 83, 99
<code>plural</code>	77, 84, 129, 164, 165, 172
<code>see</code>	6, 14, 16, 17, 68
<code>short</code>	81, 86, 110
<code>shortplural</code>	82, 86, 110
<code>symbol</code>	79, 84, 85
<code>symbolplural</code>	79, 80, 85
<code>text</code>	42, 76, 77, 83, 84, 129, 131, 164, 171
<code>\newif</code>	98, 129
<code>\newlength</code>	185
<code>\newnum</code>	8
<code>\newrobustcmd</code>	38, 39, 46–48, 57, 103, 104, 114–123, 160, 162–170, 186–192
<code>\newsym</code>	8
<code>\newterm</code>	90
<code>\newtoks</code>	110
<code>\newwrite</code>	67
<code>\NoCaseChange</code>	48–51, 162–170
<code>\noexpand</code>	10, 65, 72, 99, 100, 112, 178
<code>\nofiles</code>	74
<code>\noindent</code>	75
<code>\nopostdesc</code>	18, 19, 90
<code>\nr</code>	5, 6, 8, 10
<code>\ns@GLSxtrfull</code>	115
<code>\ns@Glsxtrfull</code>	115
<code>\ns@glxtrfull</code>	114
<code>\ns@GLSxtrfullpl</code>	117
<code>\ns@Glsxtrfullpl</code>	116
<code>\ns@glxtrfullpl</code>	116
<code>\ns@GLSxtrlong</code>	120
<code>\ns@Glsxtrlong</code>	119
<code>\ns@glxtrlong</code>	119
<code>\ns@GLSxtrlongpl</code>	123
<code>\ns@Glsxtrlongpl</code>	123
<code>\ns@glxtrlongpl</code>	122
<code>\ns@GLSxtrshort</code>	118
<code>\ns@Glsxtrshort</code>	118
<code>\ns@glxtrshort</code>	117
<code>\ns@GLSxtrshortpl</code>	121, 122
<code>\ns@Glsxtrshortpl</code>	121
<code>\ns@glxtrshortpl</code>	120
<code>\null</code>	10
<code>\number</code>	60–62
<code>\numexpr</code>	60, 62
O	
<code>\or</code>	5, 9, 16
<code>\org@glossaryentrynumbers</code>	22
P	
<code>\p@gl@s@hyp@opt</code>	41
package options:	
<code>abbreviations</code>	7
<code>accsupp</code>	9, 76
<code>acronym</code>	7
<code>automake</code>	69, 73
<code>docdef</code>	5, 16, 53, 61
<code>false</code>	16
<code>restricted</code>	6
<code>true</code>	16
<code>nonumberlist</code>	22
<code>numbers</code>	8
<code>shortcuts</code>	8
<code>all</code>	8
<code>false</code>	8
<code>none</code>	8
<code>true</code>	8
<code>style</code>	11
<code>stylemods</code>	11
<code>symbols</code>	8, 91
<code>undefaction</code>	12, 13

warn	5	\RestoreAcronyms	66
\PackageError	5, 10, 16, 20, 37–39, 47, 52–54, 61, 63, 64, 66, 68, 69, 126–129, 178	\romannumeral	185, 186, 193
\PackageWarning	6	S	
\PackageWarningNoLine	6	\s@glshyp@opt	41
\par	75, 76, 184, 185, 194	\s@glstr@enabletagging	103
\parindent	185, 194	\seenamex	14
\PassOptionsToPackage	4	\setabbreviationstyle	66, 130, 138
\preto	40	\setacronymstyle	66, 67
\printabbreviations	7	\SetGenericNewAcronym	66
\printglossaries	68, 74	\setglossarystyle	11, 179, 195
\printglossary	7, 68, 74	\setkeys	11, 40, 65, 110, 111
\printglossary options		\setlength	21, 185, 194
nonumberlist	24	\settowidth	67, 185–193
\printnoidxglossaries	74	\setupglossaries	4, 11
\printnoidxglossary	74	\sfcode	106, 184
\printnumbers	8, 91	\space	5, 17, 20, 52–55, 61, 63, 64, 66– 68, 72, 75, 106, 107, 113, 130, 184, 185, 193
\printsymbols	8, 91	\spacefactor	106, 111, 184
\ProcessOptions	178	\string	5, 16, 17, 20, 23, 24, 37, 38, 47, 48, 52– 55, 61, 63, 64, 66–69, 72–75, 94, 96, 97, 99
\ProcessOptionsX	11	\strut	179–184
\protect	48–51, 83–86, 112, 113, 129– 135, 137–150, 152, 154, 156, 158, 162–170	\subglossentry	179–184, 194
\protected@csedef	185	T	
\protected@csxdef	185	\tablehead	182, 183
\protected@edef	20, 65, 99, 112	\tabletail	182, 183
\protected@write	23, 24, 67	\tabularnewline	179–184
\providecommand	7, 24, 39, 54, 63, 67, 72, 178	\TeX	73
\ProvidesFile	176	\texorpdfstring	48–50, 170–176
\ProvidesPackage	4, 178	textcase package	159
Q		\textsc	142
\quotechar	101	\textsmaller	146
R		\texttt	72–75
\raggedright	181, 183	\the	65, 102, 112, 129–135, 137–141, 150, 152, 154, 156–158
\relax	5, 6, 8, 10, 11, 16, 17, 20, 21, 23, 41, 46, 54, 55, 63, 68, 70, 99, 100, 102, 104, 106, 110, 111, 186–195	\theindex	98
relsize package	146	\this@dialect	177
\renewcommand	5– 13, 16, 17, 19, 20, 22–25, 37, 39, 40, 42, 46, 51–54, 58, 61–69, 71, 72, 90, 92–97, 102–106, 113, 127–160, 179–184, 194, 195	\toks@	102
\renewenvironment	179–183, 193	U	
\renewglossarystyle	179–183, 193	\undef	103
\RequireGlossariesExtraLang	177	\underline	105
\RequirePackage	4, 9–11, 178	\unskip	15, 179
\reserved@a	108	\usepackage	74, 75
\reserved@b	108	V	
\reserved@d	108	\val	5, 6, 8, 10
		W	
		\warn@nomakeglossaries	68

\warn@noprintglossary	68	xindy	67
\write	54, 63, 67, 72	xkeyval package	4
		\XKV@checkchoice	24
		\XKV@plfalse	24
		\XKV@resa	24
		\XKV@sttrue	24
X			
\xcapitalisewords	92		
\xifinlist	59		