

glossaries-extra.sty v1.0: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2016-01-24

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	4
1.1 Package Initialisation and Options	4
1.2 Extra Utilities	10
1.3 Modifications to Commands Provided by glossaries	10
1.3.1 Existence Checks	11
1.3.2 Document Definitions	13
1.3.3 Existing Glossary Style Modifications	17
1.3.4 Entry Formatting, Hyperlinks and Indexing	19
1.3.5 Entry Counting	38
1.3.6 Acronym Modifications	51
1.3.7 Indexing and Displaying Glossaries	54
1.4 Integration with glossaries-accsupp	63
1.5 Categories	76
1.6 Abbreviations	96
1.6.1 Abbreviation Styles Setup	112
1.6.2 Predefined Styles (Default Font)	114
1.6.3 Predefined Styles (Small Capitals)	124
1.6.4 Predefined Styles (Fake Small Capitals)	127
1.6.5 Predefined Styles (Emphasized)	130
1.7 Using Entries in Headings	132
1.8 Multi-Lingual Support	143
Glossary	145
Change History	146
Index	151

1 Main Package Code

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2016/01/24 v1.0 (NLCT)]
```

Requires xkeyval to define package options.

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

Requires etoolbox package.

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

Has glossaries already been loaded?

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

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

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

Not already loaded, so pass options to glossaries.

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

Set the defaults.

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

Declare package options.

`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 }
```

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

```
32 \newcommand*{\glxtrwarnonexistsordo}[1]{}
```

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

```
33 \newcommand*{\glxtrundefrag}{??}
34 \newcommand*{\@glxtrundefrag}{}
```

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         \@glxtrundefrag\GlossariesExtraWarning{##1}%
41       }%
42       \renewcommand*{\glxtrwarnonexistsordo}[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         \@glxtrundefrag\PackageError{glossaries-extra}{##1}{##2}%
52       }%
53       \renewcommand*{\glxtrwarnonexistsordo}[1]{}%
54     \fi
55   }
```

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.

```
56 \define@boolkey{glossaries-extra.sty}[@glxtr]{docdef}[true]{}
```

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

```
57 \define@boolkey{glossaries-extra.sty}[@glxtr]{indexcrossrefs}[true]{%
58   \if@glxtrindexcrossrefs
59   \else
60     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
61   \fi
62 }
```

Switch off since this can increase the build time.

```
63 \@glxtrindexcrossrefsfalse
```

But allow see key to switch it on automatically.

oindexcrossrefs

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

iesExtraWarning

Allow users to suppress warnings.

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

raWarningNoLine

Allow users to suppress warnings.

```
66 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
```

```
67 \PackageWarningNoLine{glossaries-extra}{#1}}
```

```
68 \@glxtr@declareoption{nowarn}{%
```

```
69 \let\GlossariesExtraWarning\@gobble
```

```
70 \let\GlossariesExtraWarningNoLine\@gobble
```

```
71 \glxtr@doption{nowarn}%
```

```
72 }
```

glxtrabbrvtype

Glossary type for abbreviations.

```
73 \newcommand*{\glxtrabbrvtype}{\glsdefaulttype}
```

bbreviationsdef

Set by abbreviations option.

```
74 \newcommand*{\@glxtr@abbreviationsdef}{}%
```

bbreviationsdef

```
75 \newcommand*{\@glxtr@doabbreviationsdef}{%
```

```
76 \ifpackageloaded{babel}%
```

```
77 {\providecommand{\abbreviationsname}{\acronymname}}%
```

```
78 {\providecommand{\abbreviationsname}{Abbreviations}}%
```

```
79 \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
```

```
80 \renewcommand*{\glxtrabbrvtype}{abbreviations}%
```

```
81 \newcommand*{\printabbreviations}[1][1]{%
```

```
82 \printglossary[type=\glxtrabbrvtype,##1]%
```

```
83 }%
```

```
84 \disable@keys{glossaries-extra.sty}{abbreviations}%
```

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

```
85 \ifglsacronym
```

```
86 \else
```

```
87 \renewcommand*{\acronymtype}{\glxtrabbrvtype}%
```

```
88 \fi
```

```
89 }%
```

abbreviations

If abbreviations, create a new glossary type for abbreviations.

```
90 \@glxtr@declareoption{abbreviations}{%
```

```
91 \let\@glxtr@abbreviationsdef\@glxtr@doabbreviationsdef
```

```
92 }
```

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

```

93 \newcommand*\GlsXtrDefineAbbreviationShortcuts{%
94   \newcommand*\ab{\cglS}%
95   \newcommand*\abp{\cglSpl}%
96   \newcommand*\as{\glSxtrshort}%
97   \newcommand*\asp{\glSxtrshortpl}%
98   \newcommand*\al{\glSxtrlong}%
99   \newcommand*\alp{\glSxtrlongpl}%
100  \newcommand*\af{\glSxtrfull}%
101  \newcommand*\afp{\glSxtrfullpl}%
102  \newcommand*\Ab{\cGlS}%
103  \newcommand*\Abp{\cGlSpl}%
104  \newcommand*\As{\Glsxtrshort}%
105  \newcommand*\Asp{\Glsxtrshortpl}%
106  \newcommand*\Al{\Glsxtrlong}%
107  \newcommand*\Alp{\Glsxtrlongpl}%
108  \newcommand*\Af{\Glsxtrfull}%
109  \newcommand*\Afp{\Glsxtrfullpl}%
110  \newcommand*\AB{\cGLS}%
111  \newcommand*\ABP{\cGLSpl}%
112  \newcommand*\AS{\GLSxtrshort}%
113  \newcommand*\ASP{\GLSxtrshortpl}%
114  \newcommand*\AL{\GLSxtrlong}%
115  \newcommand*\ALP{\GLSxtrlongpl}%
116  \newcommand*\AF{\GLSxtrfull}%
117  \newcommand*\AFP{\GLSxtrfullpl}%
118  \newcommand*\newabbr{\newabbreviation}%

```

Disable this command after it's been used.

```

119  \let\GlsXtrDefineAbbreviationShortcuts\relax
120 }

```

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

```

121 \newcommand*\GlsXtrDefineOtherShortcuts{%
122   \newcommand*\newentry{\newglossaryentry}%
123   \ifdef\printsymbols
124     {%
125       \newcommand*\newsym{\glSxtrnewsymbol}%
126     }{}%
127   \ifdef\printnumbers
128     {%
129       \newcommand*\newnum{\glSxtrnewnumber}%
130     }{}%
131   \let\GlsXtrDefineOtherShortcuts\relax
132 }

```

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.

```
133 \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.

```
134 \define@choicekey{glossaries-extra.sty}{shortcuts}[\val\nr]%
135 {acronyms,acro,abbreviations,abbr,other,all,true,none,false}[true]{%
136   \ifcase\nr\relax % acronyms
137     \renewcommand*{\@glxtr@setupshortcuts}{%
138       \glsacrshortcutstrue
139       \DefineAcronymSynonyms
140     }%
141   \or % acro
142     \renewcommand*{\@glxtr@setupshortcuts}{%
143       \glsacrshortcutstrue
144       \DefineAcronymSynonyms
145     }%
146   \or % abbreviations
147     \renewcommand*{\@glxtr@setupshortcuts}{%
148       \GlsXtrDefineAbbreviationShortcuts
149     }%
150   \or % abbr
151     \renewcommand*{\@glxtr@setupshortcuts}{%
152       \GlsXtrDefineAbbreviationShortcuts
153     }%
154   \or % other
155     \renewcommand*{\@glxtr@setupshortcuts}{%
156       \GlsXtrDefineOtherShortcuts
157     }%
158   \or % all
159     \renewcommand*{\@glxtr@setupshortcuts}{%
160       \glsacrshortcutstrue
161       \DefineAcronymSynonyms
162       \GlsXtrDefineAbbreviationShortcuts
163       \GlsXtrDefineOtherShortcuts
164     }%
165   \or % true
166     \renewcommand*{\@glxtr@setupshortcuts}{%
167       \glsacrshortcutstrue
168       \DefineAcronymSynonyms
169       \GlsXtrDefineAbbreviationShortcuts
170       \GlsXtrDefineOtherShortcuts
171     }%
172   \else % none, false
173     \renewcommand*{\@glxtr@setupshortcuts}{}%
174   \fi
175 }
```



```

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

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

GlossaryWarning Warning text displayed in document if the external glossary file given by the argument is miss-
ing.
179 \newcommand{\glxtrNoGlossaryWarning}[1]{%
180 \@glxtr@defaultnoglossarywarning{#1}%
181 }

omissingglstext If true, suppress the text produced if the external glossary file is missing.
182 \define@choicekey{glossaries-extra.sty}{nomissingglstext}[\val\nr]%
183 {true,false}[true]{%
184 \ifcase\nr\relax % true
185 \renewcommand{\glxtrNoGlossaryWarning}[1]{%
186 \null
187 }%
188 \else % false
189 \renewcommand{\glxtrNoGlossaryWarning}[1]{%
190 \@glxtr@defaultnoglossarywarning{#1}%
191 }%
192 \fi
193 }

Pass all other options to glossaries.
194 \DeclareOptionX*{%
195 \expandafter\glxtr@doooption\expandafter{\CurrentOption}}

Process options.
196 \ProcessOptionsX

Load glossaries if not already loaded.
197 \RequirePackage{glossaries}

Load the glossaries-accsupp package if required.
198 \@glxtr@doaccsupp

Define abbreviations glossaries if required.
199 \@glxtr@abbreviationsdef
200 \let\@glxtr@abbreviationsdef\relax

Setup shortcuts if required.
201 \@glxtr@setupshortcuts

ariesextrasetup Allow user to set options after the package has been loaded. First modify \glxtr@doooption
so that it now uses \setupglossaries:
202 \renewcommand{\glxtr@doooption}[1]{\setupglossaries{#1}}%

```

Now define the user command:

```
203 \newcommand*\glossariesextrasetup}[1]{%
204   \let\@glstr@setupshortcuts\relax
205   \setkeys{glossaries-extra.sty}{#1}%
206   \@glstr@abbreviationsdef
207   \let\@glstr@abbreviationsdef\relax
208   \@glstr@setupshortcuts
209 }
```

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

```
210 \AtBeginDocument{%
211   \disable@keys{glossaries-extra.sty}{abbreviations}%
212   \def\@glstrundeftag{\glstrundeftag}%
213 }
```

1.2 Extra Utilities

rifemptyglossary

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

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

```
214 \newcommand{\glstrifemptyglossary}[3]{%
215   \ifglossaryexists{#1}%
216   {%
217     \ifcsstring{glolist@#1}{,}{#2}{#3}%
218   }%
219   {%
220     \glstrundefaction{Glossary type '#1' doesn't exist}{}%
221     #2%
222   }%
223 }
```

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.

```
224 \renewcommand{\glsdoifexists}[2]{%
225   \ifglstryexists{#1}{#2}%
226   {%
227     \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
228       has not been defined}{You need to define a glossary entry before
229       you can reference it.}%
230   }%
231 }
```

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

```
232 \renewcommand{\glsdoifnoexists}[2]{%
233   \ifglstryexists{#1}{%
234     \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
235       has already been defined}{}}{#2}%
236 }
```

`\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.

```
237 \ifdef\glsdoifexistsordo
238 {%
239   \renewcommand{\glsdoifexistsordo}[3]{%
240     \ifglstryexists{#1}{#2}%
241     {%
242       \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
243         has not been defined}{You need to define a glossary entry
244         before you can use it.}%
245       #3%
246     }%
247   }%
248 }
249 {%
250   \glstr@warnonexistsordo\glsdoifexistsordo
251   \newcommand{\glsdoifexistsordo}[3]{%
252     \ifglstryexists{#1}{#2}%
253     {%
254       \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
255         has not been defined}{You need to define a glossary entry
256         before you can use it.}%
257       #3%
258     }%
259   }%
260 }
```

`\glsdoifnoexistsordo` Similarly for `\doifglossarynoexistsordo`.

```
261 \ifdef\doifglossarynoexistsordo
262 {%
```

```

263 \renewcommand{\doifglossarynoexistsordo}[3]{%
264   \ifglossaryexists{#1}%
265   {%
266     \glstrundefaction{Glossary type ‘#1’ already exists}{}%
267     #3%
268   }%
269   {#2}%
270 }%
271 }
272 {%
273   \glstr@warnonexistsordo\doifglossarynoexistsordo
274   \newcommand{\doifglossarynoexistsordo}[3]{%
275     \ifglossaryexists{#1}%
276     {%
277       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
278       #3%
279     }%
280     {#2}%
281   }%
282 }
283

```

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

```

284 \appto\@newglossaryentryposthook{%
285   \ifdefvoid\@glo@see
286   {\csxdef{glo@\@glo@label @see}{}}%
287   {%
288     \csxdef{glo@\@glo@label @see}{\@glo@see}%
289     \@glstr@autoindexcrossrefs
290   }%
291 }
292 \appto\@gls@keymap{,{see}{see}}

```

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

```

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

```

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

```

294 \newcommand*{\glstraddallcrossrefs}{%
295   \forallglossaries{\@glo@type}%
296   {%
297     \forglsentries[\@glo@type]{\@glo@label}%
298     {%
299       \ifglused{\@glo@label}{\@glstr@addunusedxrefs{\@glo@label}}{}}%
300   }%
301 }%
302 }

```

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

```

303 \newcommand*{\@glsxtr@addunusedxrefs}[1]{%
304   \letcs{\@glo@see}{glo@glsdetoklabel{#1}@see}%
305   \ifdefvoid\@glo@see
306   {}%
307   {%
308     \expandafter\glsxtr@addunused\@glo@see\@end@glsxtr@addunused
309   }%
310 }

```

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

```

311 \newcommand*{\glsxtr@addunused}[1][]{%
312   \@glsxtr@addunused
313 }

```

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

```

314 \def\@glsxtr@addunused#1\@end@glsxtr@addunused{%
315   \@for\@glsxtr@label:=#1\do
316   {%
317     \ifglsused{\@glsxtr@label}{}%
318     {%
319       \glsadd[format=glsxtrunusedformat]{\@glsxtr@label}%
320       \glsunset{\@glsxtr@label}%
321       \@glsxtr@addunusedxrefs{\@glsxtr@label}%
322     }%
323   }%
324 }

```

`\glsxtrunusedformat`

```

325 \newcommand*{\glsxtrunusedformat}[1]{\unskip}

```

1.3.2 Document Definitions

`\makenoidxglossaries` Modify `\makenoidxglossaries` so that it automatically switches off and disables the `docdef` key.

```

326 \let\glsxtr@orgmakenoidxglossaries\makenoidxglossaries
327 \renewcommand{\makenoidxglossaries}{%
328   \glsxtr@orgmakenoidxglossaries
329   \@glsxtrdocdeffalse
330   \disable@keys{glossaries-extra.sty}{docdef}%
331 }

```

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

```

332 \renewcommand*{\gls@defdocnewglossaryentry}{%
333   \if@glsxtrdocdef

```

Since the `see` value is now saved in a field, it can be used by entries that have been defined in the document.

```

334   \let\gls@checkseeallowed\relax

```

```

335 \let\newglossaryentry\new@glossaryentry
336 \else
337 \renewcommand*{\newglossaryentry}[2]{%
338 \PackageError{glossaries-extra}{Glossary entries must
339 be \MessageBreak defined in the preamble with \MessageBreak
340 package option 'docdef=false'}{Move your glossary definitions to
341 the preamble. You can also put them in a \MessageBreak separate file
342 and load them with \string\loadglsentries.}%
343 }%
344 \fi
345 }%

```

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`

```

346 \newcommand*{\GlsXtrEnableOnTheFly}{%
347 \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
348 }

```

`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.)

```

349 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
350 \renewcommand*{\glsdetoklabel}[1]{%
351 \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
352 {%
353 \expandafter\detokenize\expandafter{##1}%
354 }%
355 {\detokenize{##1}}%
356 }%
357 \@GlsXtrEnableOnTheFly
358 }
359 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
360 \expandafter\if\glsbackslash#1%
361 #3%
362 \else
363 #4%
364 \fi
365 }

```

`sxtrstarflywarn`

```

366 \newcommand*{\glsxtrstarflywarn}{%
367 \GlossariesExtraWarning{Experimental starred version of
368 \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
369 read the warnings in the glossaries-extra user manual)}}%
370 }

```

rEnableOnTheFly

```
371 \newcommand*{\@GlsXtrEnableOnTheFly}{%
```

Don't redefine `\glsdetoklabel` if LuaTeX or XeTeX is being used, since it's mainly to allow accented characters in the label.

These definitions are all assigned the category given by:

`\glsxtrcat`

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

`\glsxtr`

```
373 \newcommand*{\glsxtr}[1] [] {%
374   \def\glsxtr@keylist{##1}%
375   \@glsxtr
376 }
```

`\@glsxtr`

```
377 \newcommand*{\@glsxtr}[2] [] {%
378   \ifglsentryexists{##2}%
379   {%
380     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
381   }%
382   {%
383     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
384       description={\nopostdesc},##1}%
385   }%
386   \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
387 }
```

`\Glsxtr`

```
388 \newcommand*{\Glsxtr}[1] [] {%
389   \def\glsxtr@keylist{##1}%
390   \@Glsxtr
391 }
```

`\@Glsxtr`

```
392 \newcommand*{\@Glsxtr}[2] [] {%
393   \ifglsentryexists{##2}%
394   {%
395     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
396   }%
397   {%
398     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
399       description={\nopostdesc},##1}%
400   }%
401   \expandafter\Gls\expandafter[\glsxtr@keylist]{##2}%
402 }
```

`\glstrpl`

```
403 \newcommand*\glstrpl[1][]{%
404 \def\glstr@keylist{##1}%
405 \@glstrpl
406 }
```

`\@glstrpl`

```
407 \newcommand*\@glstrpl[2][]{%
408 \ifglentryexists{##2}%
409 {%
410 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
411 }%
412 {%
413 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
414 description={\nopostdesc},##1}%
415 }%
416 \expandafter\glsp\expandafter[\glstr@keylist]{##2}%
417 }
```

`\Glsxtrpl`

```
418 \newcommand*\Glsxtrpl[1][]{%
419 \def\glstr@keylist{##1}%
420 \@Glsxtrpl
421 }
```

`\@Glsxtrpl`

```
422 \newcommand*\@Glsxtrpl[2][]{%
423 \ifglentryexists{##2}
424 {%
425 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
426 }%
427 {%
428 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
429 description={\nopostdesc},##1}%
430 }%
431 \expandafter\Glsp\expandafter[\glstr@keylist]{##2}%
432 }
```

`\GlsXtrWarning`

```
433 \newcommand*\GlsXtrWarning[2]{%
434 \def\@glstr@optlist{##1}%
435 \@onelevel@sanitize\@glstr@optlist
436 \GlossariesExtraWarning{The options ‘\@glstr@optlist’ have
437 been ignored for entry ‘##2’ as it has already been defined}%
438 }
```

Disable commands after the glossary:

```
439 \let\@glstr@orgprintglossary\@printglossary
```



```

440 \renewcommand\@printglossary[2]{%
441   \@glsxtr@orgprintglossary{##1}{##2}%
442   \def\@glsxtr{\@glsxtr@disabledflycommand\glsxtr}%
443   \def\@glsxtrpl{\@glsxtr@disabledflycommand\glsxtrpl}%
444   \def\@Glsxtr{\@glsxtr@disabledflycommand\Glsxtr}%
445   \def\@Glsxtrpl{\@glsxtr@disabledflycommand\Glsxtrpl}%
446 }

```

abledflycommand

```

447 \newcommand*{\@glsxtr@disabledflycommand}[1]{%
448   \PackageError{glossaries-extra}%
449   {\string##1\space can't be used after any of the \MessageBreak
450    glossaries have been displayed}%
451   {The on-the-fly commands enabled by
452    \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
453    before the glossaries. If you want to use any entries \MessageBreak
454    after any of the glossaries, you must use the standard \MessageBreak
455    method of first defining the entry and then using the \MessageBreak
456    entry with commands like \string\gls}%
457   \@glsxtr@disabledflycommand
458 }%
459 \newcommand*{\@glsxtr@disabledflycommand}[2][{}]{##2}

```

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

```

460 \let\GlsXtrEnableOnTheFly\relax
461 }
462 \@onlypreamble\GlsXtrEnableOnTheFly

```

1.3.3 Existing Glossary Style Modifications

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

```

463 \ifdef\glslistdottedwidth
464 {%
465   \ifdim\glslistdottedwidth=.5\hsize
466     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
467     \AtBeginDocument{%
468       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
469         \setlength{\glslistdottedwidth}{.5\columnwidth}%
470       \fi
471     }%
472   \fi
473 }
474 {}%

```

Similarly for `\glsdescwidth`:

`\glsdescwidth`

```

475 \ifdef\glsdescwidth

```

```

476 {%
477   \ifdim\glsdescwidth=.6\hsize
478     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
479   \AtBeginDocument{%
480     \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
481       \setlength{\glsdescwidth}{.6\columnwidth}%
482     \fi
483   }%
484 \fi
485 }
486 {}%

```

and for \glspagelistwidth:

lspagelistwidth

```

487 \ifdef\glspagelistwidth
488 {%
489   \ifdim\glspagelistwidth=.1\hsize
490     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
491   \AtBeginDocument{%
492     \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
493       \setlength{\glspagelistwidth}{.1\columnwidth}%
494     \fi
495   }%
496 \fi
497 }
498 {}%

```

aryentrynumbers Has the nonnumberlist option been used?

```

499 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
500 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
501   \glsnonnumberlistfalse
502   \renewcommand*{\glossaryentrynumbers}[1]{%
503     \GlsXtrFormatLocationList{#1}\gls@save@numberlist{#1}}%
504 \else
505   \glsnonnumberlisttrue
506   \renewcommand*{\glossaryentrynumbers}[1]{\gls@save@numberlist{#1}}%
507 \fi

```

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

```

508 \newcommand*{\GlsXtrFormatLocationList}[1]{#1}

```

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

```

509 \renewcommand*{\KV@printgloss@nonnumberlist}[1]{%
510 \XKV@plfalse
511 \XKV@sttrue
512 \XKV@checkchoice[\XKV@resa]{#1}{true,false}%

```

```

513 {%
514   \csname glsnonumberlist\XKV@resa\endcsname
515   \ifglsnonumberlist
516     \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
517   \else
518     \def\glossaryentrynumbers##1{%
519       \GlsXtrFormatLocationList{##1}%
520       \gls@save@numberlist{##1}}%
521   \fi
522 }%
523 }

```

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.

```

524 \renewcommand*{\glsentryfmt}{%
525   \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}}%
526   \glsifregular{\glslabel}%
527   {\glsgenentryfmt}%
528   {\ifglshasshort{\glslabel}{\glsxtrgenabbrvfmt}{\glsgenentryfmt}}%
529 }

```

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

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

```

530 \renewcommand{\@gls@field@link}[4][]{%
531   \glsdoifexists{#3}%
532   {%
533     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
534     \def\glscustomtext{#4}%
535     \@glsxtr@field@linkdefs
536     #1%
537     \@gls@link[#2]{#3}{#4}%
538   }%
539   \glspostlinkhook
540 }

```

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

```

541 \newcommand*{\@glxtr@field@linkdefs}{%
542   \let\glxtrifwasfirstuse\@secondoftwo
543   \let\glsifplural\@secondoftwo
544   \let\glscapscase\@firstofthree
545   \let\glsinsert\@empty
546 }

```

Redefine the field link commands that need to modify the above.

```

\@GLStext@  All uppercase version of \glstext.
547 \def\@GLStext@#1#2[#3]{%
548   \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
549   {\mfirstucMakeUppercase{\glentrytext{#2}#3}}%
550 }

\@Glstext@  First letter uppercase version.
551 \def\@Glstext@#1#2[#3]{%
552   \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
553   {\Glentrytext{#2}#3}%
554 }

\@glsfirst@  No case changing version.
555 \def\@glsfirst@#1#2[#3]{%
556   \@gls@field@link[\let\glxtrifwasfirstuse\@firstoftwo]{#1}{#2}%
557   {\glentryfirst{#2}#3}%
558 }

\@Glsfirst@  First letter uppercase version.
559 \def\@Glsfirst@#1#2[#3]{%
560   \@gls@field@link
561   [\let\glxtrifwasfirstuse\@firstoftwo
562   \let\glscapscase\@secondofthree
563   ]%
564   {#1}{#2}{\Glentryfirst{#2}#3}%
565 }

\@GLSfirst@  All uppercase version.
566 \def\@GLSfirst@#1#2[#3]{%
567   \@gls@field@link
568   [\let\glxtrifwasfirstuse\@firstoftwo
569   \let\glscapscase\@thirdofthree
570   ]%
571   {#1}{#2}{\mfirstucMakeUppercase{\glentryfirst{#2}#3}}%
572 }

\@glsplural  No case changing version.
573 \def\@glsplural#1#2[#3]{%
574   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
575   {\glentryplural{#2}#3}%
576 }

```

```

\@Glsplural    First letter uppercase version.
577 \def\@Glsplural#1#2[#3]{%
578   \@gls@field@link
579   [\let\glsifplural\@firstoftwo
580   \let\glscapscase\@secondofthree
581   ]%
582   {#1}{#2}{\Glsentryplural{#2}#3}%
583 }

\@GLSplural    All uppercase version.
584 \def\@GLSplural#1#2[#3]{%
585   \@gls@field@link
586   [\let\glsifplural\@firstoftwo
587   \let\glscapscase\@thirdofthree
588   ]%
589   {#1}{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}#3}}%
590 }

glsfirstplural@    No case changing version.
591 \def\@glsfirstplural@#1#2[#3]{%
592   \@gls@field@link
593   [\let\glsextrifwasfirstuse\@firstoftwo
594   \let\glsifplural\@firstoftwo
595   ]%
596   {#1}{#2}{\glsentryfirstplural{#2}#3}%
597 }

Glsfirstplural@    First letter uppercase version.
598 \def\@glsfirstplural@#1#2[#3]{%
599   \@gls@field@link
600   [\let\glsextrifwasfirstuse\@firstoftwo
601   \let\glsifplural\@firstoftwo
602   \let\glscapscase\@secondofthree
603   ]%
604   {#1}{#2}{\Glsentryfirstplural{#2}#3}%
605 }

Glsfirstplural@    All uppercase version.
606 \def\@glsfirstplural@#1#2[#3]{%
607   \@gls@field@link
608   [\let\glsextrifwasfirstuse\@firstoftwo
609   \let\glsifplural\@firstoftwo
610   \let\glscapscase\@thirdofthree
611   ]%
612   {#1}{#2}{\mfirstucMakeUppercase{\glsentryfirstplural{#2}#3}}%
613 }

\@Glsname@    First letter uppercase version.

```

```

614 \def\@GLSname@#1#2[#3]{%
615   \@gls@field@link
616   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryname{#2}#3}%
617 }

\@GLSname@   All uppercase version.

618 \def\@GLSname@#1#2[#3]{%
619   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
620   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryname{#2}#3}}%
621 }

\@GLSdesc@   First letter uppercase version.

622 \def\@GLSdesc@#1#2[#3]{%
623   \@gls@field@link
624   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentrydesc{#2}#3}%
625 }

\@GLSdesc@   All uppercase version.

626 \def\@GLSdesc@#1#2[#3]{%
627   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
628   {#1}{#2}{\mfirstucMakeUppercase{\Glsentrydesc{#2}#3}}%
629 }

@GLSdescplural@   No case-changing version.

630 \def\@GLSdescplural@#1#2[#3]{%
631   \@gls@field@link
632   [\let\glscapscase\@secondoftwo
633     \let\glsifplural\@firstoftwo
634   ]{#1}{#2}{\Glsentrydescplural{#2}#3}%
635 }

@GLSdescplural@   First letter uppercase version.

636 \def\@GLSdescplural@#1#2[#3]{%
637   \@gls@field@link
638   [\let\glscapscase\@secondoftwo
639     \let\glsifplural\@firstoftwo
640   ]{#1}{#2}{\Glsentrydescplural{#2}#3}%
641 }

@GLSdescplural@   All uppercase version.

642 \def\@GLSdesc@#1#2[#3]{%
643   \@gls@field@link
644   [\let\glscapscase\@thirdoftwo
645     \let\glsifplural\@firstoftwo
646   ]%
647   {#1}{#2}{\mfirstucMakeUppercase{\Glsentrydescplural{#2}#3}}%
648 }

```

```

\@GLssymbol@ First letter uppercase version.
649 \def\@GLssymbol@#1#2[#3]{%
650   \@gls@field@link
651   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentrysymbol{#2}#3}%
652 }

\@GLSsymbol@ All uppercase version.
653 \def\@GLSsymbol@#1#2[#3]{%
654   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
655   {#1}{#2}{\mfirstucMakeUppercase{\glentrysymbol{#2}#3}}%
656 }

lssymbolplural@ No case-changing version.
657 \def\@lssymbolplural@#1#2[#3]{%
658   \@gls@field@link
659   [\let\glscapscase\@secondoftwo
660     \let\glsifplural\@firstoftwo
661   ]{#1}{#2}{\glentrysymbolplural{#2}#3}%
662 }

lssymbolplural@ First letter uppercase version.
663 \def\@lssymbolplural@#1#2[#3]{%
664   \@gls@field@link
665   [\let\glscapscase\@secondoftwo
666     \let\glsifplural\@firstoftwo
667   ]{#1}{#2}{\Glsentrysymbolplural{#2}#3}%
668 }

LSsymbolplural@ All uppercase version.
669 \def\@LSsymbol@#1#2[#3]{%
670   \@gls@field@link
671   [\let\glscapscase\@thirdoftwo
672     \let\glsifplural\@firstoftwo
673   ]%
674   {#1}{#2}{\mfirstucMakeUppercase{\glentrysymbolplural{#2}#3}}%
675 }

\@Glsuseri@ First letter uppercase version.
676 \def\@Glsuseri@#1#2[#3]{%
677   \@gls@field@link
678   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuseri{#2}#3}%
679 }

\@GLSuseri@ All uppercase version.
680 \def\@GLSuseri@#1#2[#3]{%
681   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
682   {#1}{#2}{\mfirstucMakeUppercase{\glentryuseri{#2}#3}}%
683 }

```

```

\@Glsuserii@ First letter uppercase version.
684 \def\@Glsuserii@#1#2[#3]{%
685   \@gls@field@link
686   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuserii{#2}#3}%
687 }

\@Glsuserii@ All uppercase version.
688 \def\@Glsuserii@#1#2[#3]{%
689   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
690   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryuserii{#2}#3}}%
691 }

\@Glsuseriii@ First letter uppercase version.
692 \def\@Glsuseriii@#1#2[#3]{%
693   \@gls@field@link
694   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuseriii{#2}#3}%
695 }

\@Glsuseriii@ All uppercase version.
696 \def\@Glsuseriii@#1#2[#3]{%
697   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
698   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryuseriii{#2}#3}}%
699 }

\@Glsuseriv@ First letter uppercase version.
700 \def\@Glsuseriv@#1#2[#3]{%
701   \@gls@field@link
702   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuseriv{#2}#3}%
703 }

\@Glsuseriv@ All uppercase version.
704 \def\@Glsuseriv@#1#2[#3]{%
705   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
706   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryuseriv{#2}#3}}%
707 }

\@Glsuserv@ First letter uppercase version.
708 \def\@Glsuserv@#1#2[#3]{%
709   \@gls@field@link
710   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuserv{#2}#3}%
711 }

\@Glsuserv@ All uppercase version.
712 \def\@Glsuserv@#1#2[#3]{%
713   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
714   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryuserv{#2}#3}}%
715 }

```


\@Glsuservi@ First letter uppercase version.

```
716 \def\@Glsuservi@#1#2[#3]{%
717   \@gls@field@link
718   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuservi{#2}#3}%
719 }
```

\@GLSuservi@ All uppercase version.

```
720 \def\@GLSuservi@#1#2[#3]{%
721   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
722   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryuservi{#2}#3}}%
723 }
```

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

\@acrshort No case change.

```
724 \def\@acrshort#1#2[#3]{%
725   \glsoifexists{#2}%
726   {%
727     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
728     \let\glxtrifwasfirstuse\@secondoftwo
729     \let\glsifplural\@secondoftwo
730     \let\glscapscase\@firstofthree
731     \let\glsinsert\@empty
732     \def\glscustomtext{%
733       \acronymfont{\Glsaccessshort{#2}}#3%
734     }%
735     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
736   }%
737   \glspostlinkhook
738 }
```

\@Acrshort First letter uppercase.

```
739 \def\@Acrshort#1#2[#3]{%
740   \glsoifexists{#2}%
741   {%
742     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
743     \let\glxtrifwasfirstuse\@secondoftwo
744     \let\glsifplural\@secondoftwo
745     \let\glscapscase\@secondofthree
746     \let\glsinsert\@empty
747     \def\glscustomtext{%
748       \acronymfont{\Glsaccessshort{#2}}#3%
749     }%
750     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
751   }%
752   \glspostlinkhook
753 }
```

\@ACRshort All uppercase.

```
754 \def\@ACRshort#1#2[#3]{%
755   \glsdoifexists{#2}%
756   {%
757     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
758     \let\glxtrifwasfirstuse\@secondoftwo
759     \let\gl@sifplural\@secondoftwo
760     \let\glscapscase\@thirdofthree
761     \let\gl$insert\@empty
762     \def\glscustomtext{%
763       \mfirstucMakeUppercase{\acronymfont{\gl@saccesssshort{#2}}#3}%
764     }%
765     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
766   }%
767   \glspostlinkhook
768 }
```

\@acrshortpl No case change.

```
769 \def\@acrshortpl#1#2[#3]{%
770   \glsdoifexists{#2}%
771   {%
772     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
773     \let\glxtrifwasfirstuse\@secondoftwo
774     \let\gl@sifplural\@firstoftwo
775     \let\glscapscase\@firstofthree
776     \let\gl$insert\@empty
777     \def\glscustomtext{%
778       \acronymfont{\gl@saccesssshortpl{#2}}#3%
779     }%
780     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
781   }%
782   \glspostlinkhook
783 }
```

\@Acrshortpl First letter uppercase.

```
784 \def\@Acrshortpl#1#2[#3]{%
785   \glsdoifexists{#2}%
786   {%
787     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
788     \let\glxtrifwasfirstuse\@secondoftwo
789     \let\gl@sifplural\@firstoftwo
790     \let\glscapscase\@secondofthree
791     \let\gl$insert\@empty
792     \def\glscustomtext{%
793       \acronymfont{\Glsaccesssshortpl{#2}}#3%
794     }%
795     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
796   }%
797   \glspostlinkhook
```

798 }

\@ACRshortpl All uppercase.

```
799 \def\@ACRshortpl#1#2[#3]{%
800   \glsdoifexists{#2}%
801   {%
802     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
803     \let\glxtrifwasfirstuse\@secondoftwo
804     \let\glsifplural\@firstoftwo
805     \let\glscapscase\@thirdofthree
806     \let\glsinsert\@empty
807     \def\glscustomtext{%
808       \mfirstucMakeUppercase{\acronymfont{\glsaccessshortpl{#2}}#3}%
809     }%
810     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
811   }%
812   \glspostlinkhook
813 }
```

\@acrlong No case change.

```
814 \def\@acrlong#1#2[#3]{%
815   \glsdoifexists{#2}%
816   {%
817     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
818     \let\glxtrifwasfirstuse\@secondoftwo
819     \let\glsifplural\@secondoftwo
820     \let\glscapscase\@firstofthree
821     \let\glsinsert\@empty
822     \def\glscustomtext{%
823       \acronymfont{\glsaccesslong{#2}}#3%
824     }%
825     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
826   }%
827   \glspostlinkhook
828 }
```

\@Acrlong First letter uppercase.

```
829 \def\@Acrlong#1#2[#3]{%
830   \glsdoifexists{#2}%
831   {%
832     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
833     \let\glxtrifwasfirstuse\@secondoftwo
834     \let\glsifplural\@secondoftwo
835     \let\glscapscase\@secondofthree
836     \let\glsinsert\@empty
837     \def\glscustomtext{%
838       \acronymfont{\Glsaccesslong{#2}}#3%
839     }%
840     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
841   }
```

```

841 }%
842 \glspostlinkhook
843 }

```

\@ACRlong All uppercase.

```

844 \def\@ACRlong#1#2[#3]{%
845   \glsdoifexists{#2}%
846   {%
847     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
848     \let\glxtrifwasfirstuse\@secondoftwo
849     \let\gl@sifplural\@secondoftwo
850     \let\glscapscase\@thirdofthree
851     \let\gl$insert\@empty
852     \def\glscustomtext{%
853       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslong{#2}}#3}%
854     }%
855     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
856   }%
857   \glspostlinkhook
858 }

```

\@acrlongpl No case change.

```

859 \def\@acrlongpl#1#2[#3]{%
860   \glsdoifexists{#2}%
861   {%
862     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
863     \let\glxtrifwasfirstuse\@secondoftwo
864     \let\gl@sifplural\@firstoftwo
865     \let\glscapscase\@firstofthree
866     \let\gl$insert\@empty
867     \def\glscustomtext{%
868       \acronymfont{\gl@saccesslongpl{#2}}#3%
869     }%
870     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
871   }%
872   \glspostlinkhook
873 }

```

\@Acrlongpl First letter uppercase.

```

874 \def\@Acrlongpl#1#2[#3]{%
875   \glsdoifexists{#2}%
876   {%
877     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
878     \let\glxtrifwasfirstuse\@secondoftwo
879     \let\gl@sifplural\@firstoftwo
880     \let\glscapscase\@secondofthree
881     \let\gl$insert\@empty
882     \def\glscustomtext{%
883       \acronymfont{\Glsaccesslongpl{#2}}#3%

```

```

884 }%
885 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
886 }%
887 \glspostlinkhook
888 }

```

\@ACRlongpl All uppercase.

```

889 \def\@ACRlongpl#1#2[#3]{%
890 \glsdoifexists{#2}%
891 {%
892 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
893 \let\glstrifwasfirstuse\@secondoftwo
894 \let\glsifplural\@firstoftwo
895 \let\glscapscase\@thirdofthree
896 \let\glsinsert\@empty
897 \def\glscustomtext{%
898 \mfirstucMakeUppercase{\acronymfont{\glsaccesslongpl{#2}}#3}%
899 }%
900 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
901 }%
902 \glspostlinkhook
903 }

```

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

\glsaddkey

```

904 \renewcommand*{\@glsaddkey}[7]{%
905 \key@ifundefined{glossentry}{#1}%
906 {%
907 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
908 \appto\@gls@keymap{,{#1}{#1}}%
909 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
910 \appto\@newglossaryentryposthook{%
911 \letcs{\@glo@tmp}{@glo@#1}%
912 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
913 }%
914 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
915 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

916 \ifcsdef{@gls@user@#1@}%
917 {%
918 \PackageError{glossaries}%
919 {Can't define '\string#5' as helper command
920 '\expandafter\string\csname @gls@user@#1@\endcsname' already
921 exists}%
922 }%
923 }%
924 {%

```

```

925 \expandafter\newcommand\expandafter*\expandafter
926 {\csname @gls@user@#1\endcsname}[2][\%
927 \new@ifnextchar[
928 {\csuse{@gls@user@#1@}{##1}{##2}}%
929 {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
930 \csdef{@gls@user@#1@}##1##2[##3]{%
931 \@gls@field@link{##1}{##2}{#3{##2}##3}%
932 }%
933 \newrobustcmd*{#5}{%
934 \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
935 }%

```

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

```

936 \ifcsdef{@Gls@user@#1@}%
937 {%
938 \PackageError{glossaries}%
939 {Can't define '\string#6' as helper command
940 '\expandafter\string\csname @Gls@user@#1\endcsname' already
941 exists}%
942 }%
943 }%
944 {%
945 \expandafter\newcommand\expandafter*\expandafter
946 {\csname @Gls@user@#1\endcsname}[2][\%
947 \new@ifnextchar[
948 {\csuse{@Gls@user@#1@}{##1}{##2}}%
949 {\csuse{@Gls@user@#1@}{##1}{##2}[]}}%
950 \csdef{@Gls@user@#1@}##1##2[##3]{%
951 \@gls@field@link[\let\glscaps@case\@secondofthree]%
952 {##1}{##2}{#4{##2}##3}%
953 }%
954 \newrobustcmd*{#6}{%
955 \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
956 }%

```

Finally the all caps version (modified):

```

957 \ifcsdef{@GLS@user@#1@}%
958 {%
959 \PackageError{glossaries}%
960 {Can't define '\string#7' as helper command
961 '\expandafter\string\csname @GLS@user@#1\endcsname' already
962 exists}%
963 }%
964 }%
965 {%
966 \expandafter\newcommand\expandafter*\expandafter
967 {\csname @GLS@user@#1\endcsname}[2][\%
968 \new@ifnextchar[
969 {\csuse{@GLS@user@#1@}{##1}{##2}}%
970 {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%

```

```

971 \csdef{@GLS@user@#1@}##1##2[##3]{%
972 \@gls@field@link[\let\glscapscase\@thirdofthree]%
973 {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}}%
974 }%
975 \newrobustcmd*{#7}{%
976 \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
977 }%
978 }%
979 {%
980 \PackageError{glossaries-extra}{Key ‘#1’ already exists}{}%
981 }%
982 }

```

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

```

983 \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.

```

984 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
985 \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.

```

986 \ifglsused{\glslabel}%
987 {\let\glsxtrifwasfirstuse\@secondoftwo}
988 {\let\glsxtrifwasfirstuse\@firstoftwo}%

```

Store the category label for convenience.

```

989 \edef\glscategorylabel{\glscategory{\glslabel}}%
990 \ifglsused{\glslabel}%
991 {%
992 \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
993 {\KV@glslink@hyperfalse}}}%
994 }%
995 {%
996 \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
997 {\KV@glslink@hyperfalse}}}%
998 }%
999 \glslinkcheckfirsthyperhook
1000 }

```

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.

```

1001 \ifdef\do@glstdisablehyperinlist
1002 {%
1003 \let\@glsxtr@do@glstdisablehyperinlist\do@glstdisablehyperinlist
1004 \renewcommand*{\do@glstdisablehyperinlist}{%

```

```

1005 \glsxtr@do@gl:disablehyperinlist
1006 \glsifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}%
1007 }
1008 }
1009 {}

```

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

```

1010 \define@boolkey{glslink}{noindex}[true]{}
1011 \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`

```

1012 \ifdef\@gls@setdefault@glslink@opts
1013 {
1014 \renewcommand*{\@gls@setdefault@glslink@opts}{%
1015 \KV@glslink@noindexfalse
1016 }
1017 }
1018 {

```

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

```

1019 \newcommand*{\@gls@setdefault@glslink@opts}{%
1020 \KV@glslink@noindexfalse
1021 }
1022 \preto\do@gl:disablehyperinlist{\@gls@setdefault@glslink@opts}
1023 }

```

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

```

1024 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
1025 \renewcommand*{\@gls@setdefault@glslink@opts}{\setkeys{glslink}{#1}}%
1026 }

```

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

```

1027 \newcommand*{\glstrifindexing}[2]{%
1028 \ifKV@glslink@noindex #2\else #1\fi
1029 }

```

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

```

1030 \renewcommand*{\glswriteentry}[2]{%
1031 \glstrifindexing
1032 {%
1033 \ifglindexonlyfirst
1034 \ifglused{#1}
1035 {\glxtrdoautoindexname{#1}{dualindex}}%
1036 {#2}%
1037 \else
1038 \glsifattribute{#1}{indexonlyfirst}{true}%
1039 {\ifglused{#1}

```



```

1040      {\glstrdoautoindexname{#1}{dualindex}}}%
1041      {#2}}}%
1042      {#2}}%
1043      \fi
1044    }%
1045    {}%
1046  }

```

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

```

1047 \appto@@do@@wrglossary{\@glstrdo@@wrindex
1048   \glstrdowrglossaryhook{\@gls@label}}%
1049 }

```

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

Similarly for the “noidx” version:

`s@noidxglossary`

```

1050 \appto\gls@noidxglossary{\@glstrdo@@wrindex
1051   \glstrdowrglossaryhook{\@gls@label}}%
1052 }

```

`xtr@do@@wrindex`

```

1053 \newcommand*{\@glstrdo@@wrindex}{%
1054   \glstrdoautoindexname{\@gls@label}{dualindex}}%
1055 }

```

`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.)

```

1056 \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.

```

1057 \newcommand*{\@gls@alt@hyp@opt}[1]{%
1058   \let\glslinkvar\@firstofthree
1059   \let\@gls@hyp@opt@cs#1\relax
1060   \@ifstar{\s@gls@hyp@opt}%
1061   {\@ifnextchar+%
1062     {\@firstoftwo{\p@gls@hyp@opt}}%
1063     {%
1064       \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
1065       {\@firstoftwo{\@alt@gls@hyp@opt}}%
1066       {#1}}%
1067   }%
1068 }%
1069 }

```

`alt@glshyp@opt` User version

```

1070 \newcommand*{\@alt@glshyp@opt}[1] [] {%
1071 \let\glslinkvar\@firstofthree
1072 \expandafter\@glshyp@opt@cs\expandafter[\@glshyp@opt@keys,#1]}

```

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

```

1073 \newcommand*{\@glshyp@opt@char}{}

```

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

```

1074 \newcommand*{\@glshyp@opt@keys}{}

```

`rSetAltModifier`

```

1075 \newcommand*{\GlsXtrSetAltModifier}[2] {%
1076 \let\@glshyp@opt\@glshyp@opt
1077 \def\@glshyp@opt@char{#1}%
1078 \def\@glshyp@opt@keys{#2}%
1079 }

```

`\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`.

```

1080 \renewcommand*{\glsdohyperlink}[2] {%
1081 \hyperlink{#1}{\glxtrprotectlinks#2}}

```

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

```

1082 \ifundef\glsdonohyperlink
1083 {%
1084 \renewcommand{\gl:disablehyper}{%
1085 \KV@glslink@hyperfalse
1086 \let\@glslink\glsdonohyperlink
1087 \let\@glstarget\@secondoftwo
1088 }
1089 }
1090 {}

```

`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.

```

1091 \def\gl:sdonohyperlink#1#2{\glxtrprotectlinks #2}

```

Reset `\@glslink` with patched versions:

```

1092 \ifcsundef{hyperlink}%
1093 {%

```

```

1094 \let\@glslink\glsdonohyperlink
1095 }%
1096 {%
1097 \let\@glslink\glsdohyperlink
1098 }

```

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

```

1099 \newcommand*{\glsxtrprotectlinks}{%
1100 \KV@glslink@hyperfalse
1101 \KV@glslink@noindextrue
1102 \let\@gls@\@glsxtr@p@text@
1103 \let\@Gls@\@Glsxtr@p@text@
1104 \let\@GLS@\@GLSxtr@p@text@
1105 \let\@glspl@\@glsxtr@p@plural@
1106 \let\@Glspl@\@Glsxtr@p@plural@
1107 \let\@GLSpl@\@GLSxtr@p@plural@
1108 \let\@glsxtrshort\@glsxtr@p@short@
1109 \let\@Glsxtrshort\@Glsxtr@p@short@
1110 \let\@GLSxtrshort\@GLSxtr@p@short@
1111 \let\@glsxtrlong\@glsxtr@p@long@
1112 \let\@Glsxtrlong\@Glsxtr@p@long@
1113 \let\@GLSxtrlong\@GLSxtr@p@long@
1114 \let\@glsxtrshortpl\@glsxtr@p@shortpl@
1115 \let\@Glsxtrshortpl\@Glsxtr@p@shortpl@
1116 \let\@GLSxtrshortpl\@GLSxtr@p@shortpl@
1117 \let\@glsxtrlongpl\@glsxtr@p@longpl@
1118 \let\@Glsxtrlongpl\@Glsxtr@p@longpl@
1119 \let\@GLSxtrlongpl\@GLSxtr@p@longpl@
1120 \let\@acrshort\@glsxtr@p@acrshort@
1121 \let\@Acrshort\@Glsxtr@p@acrshort@
1122 \let\@ACRshort\@GLSxtr@p@acrshort@
1123 \let\@acrshortpl\@glsxtr@p@acrshortpl@
1124 \let\@Acrshortpl\@Glsxtr@p@acrshortpl@
1125 \let\@ACRshortpl\@GLSxtr@p@acrshortpl@
1126 \let\@acrlong\@glsxtr@p@acrlong@
1127 \let\@Acrlong\@Glsxtr@p@acrlong@
1128 \let\@ACRlong\@GLSxtr@p@acrlong@
1129 \let\@acrlongpl\@glsxtr@p@acrlongpl@
1130 \let\@Acrlongpl\@Glsxtr@p@acrlongpl@
1131 \let\@ACRlongpl\@GLSxtr@p@acrlongpl@
1132 }

```

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

`@glsxtr@p@text@`

```

1133 \def\@glsxtr@p@text@#1#2[#3]{\@gls{text@{#1}{#2}[#3]}}

```

`@Glsxtr@p@text@`

```

1134 \def\@Glsxtr@p@text@#1#2[#3]{\@Gls{text@{#1}{#2}[#3]}}

```

```

@GLSxtr@p@text@
1135 \def\@GLSxtr@p@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}}

lsxtr@p@plural@
1136 \def\@glsxtr@p@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}}

lsxtr@p@plural@
1137 \def\@Glsxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}}

LSxtr@p@plural@
1138 \def\@GLSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}}

glsxtr@p@short@
1139 \def\@glsxtr@p@short@#1#2[#3]{%
1140 {%
1141   \glsssetabbrvfmt{\glscategory{#2}}}%
1142   \glsabbrvfont{\glsentryshort{#2}}#3}%
1143 }%
1144 }

Glsxtr@p@short@
1145 \def\@Glsxtr@p@short@#1#2[#3]{%
1146 {%
1147   \glsssetabbrvfmt{\glscategory{#2}}}%
1148   \glsabbrvfont{\Glsentryshort{#2}}#3}%
1149 }%
1150 }

GLSxtr@p@short@
1151 \def\@GLSxtr@p@short@#1#2[#3]{%
1152 {%
1153   \glsssetabbrvfmt{\glscategory{#2}}}%
1154   \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshort{#2}}#3}%
1155 }%
1156 }

sxtr@p@shortpl@
1157 \def\@glsxtr@p@shortpl@#1#2[#3]{%
1158 {%
1159   \glsssetabbrvfmt{\glscategory{#2}}}%
1160   \glsabbrvfont{\glsentryshortpl{#2}}#3}%
1161 }%
1162 }

sxtr@p@shortpl@
1163 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
1164 {%
1165   \glsssetabbrvfmt{\glscategory{#2}}}%

```

```

1166 \glsabbrvfont{\Glsentryshortpl{#2}}#3%
1167 }%
1168 }

Sxtr@p@shortpl@
1169 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
1170 {%
1171 \glssetabbrvfmt{\glscategory{#2}}%
1172 \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshortpl{#2}}#3}%
1173 }%
1174 }

@glsxtr@p@long@
1175 \def\@glsxtr@p@long@#1#2[#3]{\{\glsentrylong{#2}#3}}

@Glsxtr@p@long@
1176 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}

@GLSxtr@p@long@
1177 \def\@GLSxtr@p@long@#1#2[#3]{%
1178 {\mfirstucMakeUppercase{\glsentrylong{#2}#3}}}

lsxtr@p@longpl@
1179 \def\@glsxtr@p@longpl@#1#2[#3]{\{\glsentrylongpl{#2}#3}}

lsxtr@p@longpl@
1180 \def\@Glsxtr@p@longpl@#1#2[#3]{\{\Glsentrylongpl{#2}#3}}

LSxtr@p@longpl@
1181 \def\@GLSxtr@p@longpl@#1#2[#3]{%
1182 {\mfirstucMakeUppercase{\glsentrylongpl{#2}#3}}}

xtr@p@acrshort@
1183 \def\@glsxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\glsentryshort{#2}}#3}}

xtr@p@acrshort@
1184 \def\@Glsxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
1185 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
1186 {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
1187 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
1188 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\Glsentryshortpl{#2}}#3}}

```

```

r@p@acrshortpl@
1189 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
1190   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}#3}}}

s@xtr@p@acrlong@
1191 \def\@GLSxtr@p@acrlong@#1#2[#3]{\glsentrylong{#2}#3}}

s@xtr@p@acrlong@
1192 \def\@GLSxtr@p@acrlong@#1#2[#3]{\Glsentrylong{#2}#3}}

S@xtr@p@acrlong@
1193 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
1194   {\mfirstucMakeUppercase{\glsentrylong{#2}#3}}}

tr@p@acrlongpl@
1195 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{\glsentrylongpl{#2}#3}}

tr@p@acrlongpl@
1196 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{\Glsentrylongpl{#2}#3}}

tr@p@acrlongpl@
1197 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
1198   {\mfirstucMakeUppercase{\glsentrylongpl{#2}#3}}}

```

1.3.5 Entry Counting

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

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

```

\@glsunset   Global unset.
1199 \renewcommand*{\@glsunset}[1]{%
1200   \@glsunset{#1}%
1201   \glsxtrpostunset{#1}%
1202 }%

glsxtrpostunset
1203 \newcommand*{\glsxtrpostunset}[1]{%

\@glslocalunset   Local unset.
1204   \renewcommand*{\@glslocalunset}[1]{%
1205     \@glslocalunset{#1}%
1206     \glsxtrpostlocalunset{#1}%
1207   }%

```

rpostlocalunset

```
1208 \newcommand*{\glxtrpostlocalunset}[1]{}
```

\@glsreset Global reset.

```
1209 \renewcommand*{\@glsreset}[1]{%
1210   \@glsreset{#1}%
1211   \glxtrpostreset{#1}%
1212 }%
```

glxtrpostreset

```
1213 \newcommand*{\glxtrpostreset}[1]{}
```

\@glslocalreset Local reset.

```
1214 \renewcommand*{\@glslocalreset}[1]{%
1215   \@glslocalreset{#1}%
1216   \glxtrpostlocalreset{#1}%
1217 }%
```

rpostlocalreset

```
1218 \newcommand*{\glxtrpostlocalreset}[1]{}
```

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

```
1219 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

 Enable entry counting:

```
1220   \glsenableentrycount
```

 Redefine \gls etc:

```
1221   \renewcommand*{\gls}{\cglsl}%
1222   \renewcommand*{\Gls}{\cGls}%
1223   \renewcommand*{\glspl}{\cglspl}%
1224   \renewcommand*{\Glspl}{\cGlspl}%
1225   \renewcommand*{\GLS}{\cGLS}%
1226   \renewcommand*{\GLSpl}{\cGLSpl}%
```

 Set the entrycount attribute:

```
1227   \@glxtr@setentrycountunsetattr{#1}{#2}%
```

 In case this command is used again:

```
1228   \let\GlsXtrEnableEntryCounting\@glxtr@setentrycountunsetattr
1229   \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
1230     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
1231       can't be used with \string\GlsXtrEnableEntryCounting}%
1232     {Use one or other but not both commands}}%
1233 }
```

ycountunsetattr

```
1234 \newcommand*{\@glxtr@setentrycountunsetattr}[2]{%
```

```

1235 \@for\@glxtr@cat:=#1\do
1236 {%
1237   \ifdefempty{\@glxtr@cat}{}%
1238   {%
1239     \glsssetcategoryattribute{\@glxtr@cat}{entrycount}{#2}%
1240   }%
1241 }%
1242 }

```

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

nableentrycount

```

1243 \renewcommand*\@glsenableentrycount{%
  Enable new fields:
1244   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
  Just in case the user has switched on the docdef option.
1245   \renewcommand*\@gls@defdocnewglossaryentry{%
1246     \renewcommand*\newglossaryentry[2]{%
1247       \PackageError{glossaries}{\string\newglossaryentry\space
1248         may only be used in the preamble when entry counting has
1249         been activated}{If you use \string\glsenableentrycount\space
1250         you must place all entry definitions in the preamble not in
1251         the document environment}%
1252     }%
1253   }%

```

New commands to access new fields:

```

1254   \newcommand*\@glsentrycurrcount[1]{%
1255     \ifcsundef{glo@\@glsdetoklabel{##1}@currcount}%
1256     {0}{\@gls@entry@field{##1}{currcount}}%
1257   }%
1258   \newcommand*\@glsentryprevcount[1]{%
1259     \ifcsundef{glo@\@glsdetoklabel{##1}@prevcount}%
1260     {0}{\@gls@entry@field{##1}{prevcount}}%
1261   }%

```

Adjust post unset and reset:

```

1262   \let\@glxtr@entrycount@org@unset\glxtrpostunset
1263   \renewcommand*\@glxtrpostunset[1]{%
1264     \@glxtr@entrycount@org@unset{##1}%
1265     \@gls@increment@currcount{##1}%
1266   }%
1267   \let\@glxtr@entrycount@org@localunset\glxtrpostlocalunset
1268   \renewcommand*\@glxtrpostlocalunset[1]{%
1269     \@glxtr@entrycount@org@localunset{##1}%
1270     \@gls@local@increment@currcount{##1}%
1271   }%
1272   \let\@glxtr@entrycount@org@reset\glxtrpostreset
1273   \renewcommand*\@glxtrpostreset[1]{%

```



```

1274 \@glxtr@entrycount@org@reset{##1}%
1275 \csgdef{glo@glsetoklabel{##1}@currcount}{0}%
1276 }%
1277 \let\@glxtr@entrycount@org@localreset\glxtrpostlocalreset
1278 \renewcommand*{\@glxtrpostlocalreset}[1]{%
1279 \@glxtr@entrycount@org@localreset{##1}%
1280 \csdef{glo@glsetoklabel{##1}@currcount}{0}%
1281 }%

```

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

```

1282 \let\@cgl@\@cgl@
1283 \let\@cgl@sp\@cgl@sp
1284 \let\@cgl@S\@cgl@S
1285 \let\@cgl@sp\@cgl@sp
1286 \let\@cgl@S\@cgl@S
1287 \let\@cgl@sp\@cgl@sp

```

The rest is as the original definition.

```

1288 \AtEndDocument{\@gl@write@entrycounts}%
1289 \renewcommand*{\@gl@entry@count}[2]{%
1290 \csgdef{glo@glsetoklabel{##1}@prevcount}{##2}%
1291 }%
1292 \let\@gl@enable@entrycount\relax
1293 \renewcommand*{\@gl@enable@entryunitcount}{%
1294 \PackageError{glossaries-extra}{\string\@gl@enable@entryunitcount\space
1295 can't be used with \string\@gl@enable@entrycount}%
1296 {Use one or other but not both commands}%
1297 }%
1298 }

```

`\@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.

```

1299 \renewcommand*{\@gl@write@entrycounts}{%
1300 \immediate\write\@auxout
1301 {\string\providecommand*{\string\@gl@entry@count}[2]{}}%
1302 \count@=0\relax
1303 \forall@gl@sentries{\@gl@sentry}{%
1304 \gl@hasattribute{\@gl@sentry}{entrycount}%
1305 {%
1306 \ifgl@sused{\@gl@sentry}%
1307 {%
1308 \immediate\write\@auxout
1309 {\string\@gl@entry@count{\@gl@sentry}{\gl@sentrycurrcount{\@gl@sentry}}}%
1310 }%
1311 }%
1312 \advance\count@ by \@ne
1313 }%
1314 }%
1315 }%

```

```

1316 \ifnum\count@=0
1317   \GlossariesExtraWarningNoLine{Entry counting has been enabled
1318   \MessageBreak with \string\glsenableentrycount\space but the
1319   \MessageBreak attribute ‘entrycount’ hasn’t
1320   \MessageBreak been assigned to any of the defined
1321   \MessageBreak entries}%
1322 \fi
1323 }

```

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

```

1324 \newcommand*{\glxtrifcounttrigger}[3]{%
1325   \glshasattribute{#1}{entrycount}%
1326   {%
1327     \ifnum\gl Sentryprevcount{#1}>\gl sgetattribute{#1}{entrycount}\relax
1328     #3%
1329   \else
1330     #2%
1331   \fi
1332 }%
1333 {#3}%
1334 }

```

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

`\@@cgl`s@

```

1335 \def\@@cgl s@#1#2[#3]{%
1336   \glxtrifcounttrigger{#2}%
1337   {%
1338     \cgl sformat{#2}{#3}%
1339     \gl sunset{#2}%
1340   }%
1341   {%
1342     \@gl s@{#1}{#2}[#3]%
1343   }%
1344 }%

```

`\@@cgl`s@

```

1345 \def\@@cgl spl@#1#2[#3]{%
1346   \glxtrifcounttrigger{#2}%
1347   {%
1348     \cgl splformat{#2}{#3}%
1349     \gl sunset{#2}%
1350   }%
1351   {%
1352     \@gl spl@{#1}{#2}[#3]%

```

```
1353 }%
1354 }%
```

\@@cGls@

```
1355 \def\@@cGls@#1#2[#3]{%
1356   \glxtrifcounttrigger{#2}%
1357   {%
1358     \cGlsformat{#2}{#3}%
1359     \glunset{#2}%
1360   }%
1361   {%
1362     \@Gls@{#1}{#2}[#3]%
1363   }%
1364 }%
```

\@@cGlspl@

```
1365 \def\@@cGlspl@#1#2[#3]{%
1366   \glxtrifcounttrigger{#2}%
1367   {%
1368     \cGlsplformat{#2}{#3}%
1369     \glunset{#2}%
1370   }%
1371   {%
1372     \@Glspl@{#1}{#2}[#3]%
1373   }%
1374 }%
```

\@@cGLS@

```
1375 \def\@@cGLS@#1#2[#3]{%
1376   \glxtrifcounttrigger{#2}%
1377   {%
1378     \cGLSformat{#2}{#3}%
1379     \glunset{#2}%
1380   }%
1381   {%
1382     \@GLS@{#1}{#2}[#3]%
1383   }%
1384 }%
```

\@@cGLSpl@

```
1385 \def\@@cGLSpl@#1#2[#3]{%
1386   \glxtrifcounttrigger{#2}%
1387   {%
1388     \cGLSplformat{#2}{#3}%
1389     \glunset{#2}%
1390   }%
1391   {%
1392     \@GLSpl@{#1}{#2}[#3]%
1393   }%
```

```

1394 }%
1395 %
1396 % Remove default warnings from \cs{cgl}s} etc so that it can be used
1397 % interchangeable with \cs{gl}s} etc.
1398 %\begin{macro}{\@cgl@s@}
1399 %    \begin{macrocode}
1400 \def\@cgl@s@#1#2[#3]{\@gl@s@{#1}{#2}[#3]}

```

\@cGl@s@

```

1401 \def\@cGl@s@#1#2[#3]{\@Gl@s@{#1}{#2}[#3]}

```

\@cgl spl@

```

1402 \def\@cgl spl@#1#2[#3]{\@gl spl@{#1}{#2}[#3]}

```

\@cGL spl@

```

1403 \def\@cGL spl@#1#2[#3]{\@GL spl@{#1}{#2}[#3]}

```

Add all upper case versions not provided by glossaries.

\cGLS

```

1404 \newrobustcmd*{\cGLS}{\@gl@s@hyp@opt\@cGLS}

```

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

```

1405 \newcommand*{\@cGLS}[2][\%
1406   \new@ifnextchar[\@cGLS@{#1}{#2}]{\@cGLS@{#1}{#2}[]}%
1407 }

```

\@cGLS@

```

1408 \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.

```

1409 \newcommand*{\cGLSformat}[2]{\%
1410   \expandafter\mfirstucMakeUppercase\expandafter{\cgl sformat{#1}{#2}}%
1411 }

```

\cGLSpl

```

1412 \newrobustcmd*{\cGLSpl}{\@gl@s@hyp@opt\@cGLSpl}

```

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

```

1413 \newcommand*{\@cGLSpl}[2][\%
1414   \new@ifnextchar[\@cGLSpl@{#1}{#2}]{\@cGLSpl@{#1}{#2}[]}%
1415 }

```

\@cGLSpl@

```

1416 \def\@cGLSpl@#1#2[#3]{\@GLSpl@{#1}{#2}[#3]}

```

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

```
1417 \newcommand*{\cGLSplformat}[2]{%
1418   \expandafter\mfirstuc\MakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
1419 }
```

Modify the trigger formats to check for the regular attribute.

`\cglformat`

```
1420 \renewcommand*{\cglformat}[2]{%
1421   \glsifregular{#1}
1422   {\glsentryfirst{#1}}%
1423   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
1424 }
```

`\cGlsformat`

```
1425 \renewcommand*{\cGlsformat}[2]{%
1426   \glsifregular{#1}
1427   {\Glsentryfirst{#1}}%
1428   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
1429 }
```

`\cglsplformat`

```
1430 \renewcommand*{\cglsplformat}[2]{%
1431   \glsifregular{#1}
1432   {\glsentryfirstplural{#1}}%
1433   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
1434 }
```

`\cGlsplformat`

```
1435 \renewcommand*{\cGlsplformat}[2]{%
1436   \glsifregular{#1}
1437   {\Glsentryfirstplural{#1}}%
1438   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}}%#2%
1439 }
```

New code similar to above for unit counting.

`defunitcounters`

```
1440 \newcommand*{\@newglossaryentry@defunitcounters}{%
1441   \edef\@glo@countunit{\csuse{@glsxtr@categoryattr@{\@glo@category @unitcount}}}%
1442   \ifdefvoid\@glo@countunit
1443   {}%
1444   {%
1445     \@glsxtr@ifunitcounter{\@glo@countunit}%
1446     {}%
1447     {\expandafter\@glsxtr@addunitcounter\expandafter{\@glo@countunit}}%
1448   }%
1449 }
```

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

```
1450 \newcommand*{\@glxtr@unitcountlist}{}
```

@addunitcounter

```
1451 \newcommand*{\@glxtr@addunitcounter}[1]{%
1452 \listadd{\@glxtr@unitcountlist}{#1}%
1453 \ifcsundef{glxtr@theunit@#1}
1454 {%
1455 \ifcsdef{theH#1}%
1456 {\csdef{glxtr@theunit@#1}{\csuse{theH#1}}}%
1457 {\csdef{glxtr@theunit@#1}{\csuse{the#1}}}%
1458 }%
1459 }%
1460 }
```

r@ifunitcounter

```
1461 \newcommand*{\@glxtr@ifunitcounter}[3]{%
1462 \xifinlist{#1}{\@glxtr@unitcountlist}{#2}{#3}%
1463 }
```

urrentunitcount

```
1464 \newcommand*\@glxtr@currentunitcount[1]{%
1465 glo@\glsdetoklabel{#1}@currunit@glsggetattribute{#1}{unitcount}.%
1466 \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}%
1467 }
```

eviousunitcount

```
1468 \newcommand*\@glxtr@previousunitcount[1]{%
1469 glo@\glsdetoklabel{#1}@prevunit@glsggetattribute{#1}{unitcount}.%
1470 \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}%
1471 }
```

t@currunitcount

```
1472 \newcommand*{\@glx@increment@currunitcount}[1]{%
1473 \glshasattribute{#1}{unitcount}%
1474 {%
1475 \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
1476 \ifcsundef{\@glxtr@csname}%
1477 {%
1478 \csgdef{\@glxtr@csname}{1}%
1479 \listcsxadd
1480 {glo@\glsdetoklabel{#1}@unitlist}%
1481 {\glsggetattribute{#1}{unitcount}.%
1482 \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}%
1483 }%
1484 }%
1485 {%
1486 \csxdef{\@glxtr@csname}%

```

```

1487     {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
1488   }%
1489 }%
1490 {}%
1491 }

```

t@currunitcount

```

1492 \newcommand*{\@glx@local@increment@currunitcount}[1]{%
1493   \glshasattribute{#1}{unitcount}%
1494   {%
1495     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
1496     \ifcsundef{\@glxtr@csname}%
1497     {%
1498       \csdef{\@glxtr@csname}{1}%
1499       \listcseadd
1500       {glo@\glsdetoklabel{#1}@unitlist}%
1501       {\glsggetattribute{#1}{unitcount}.%
1502        \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}}%
1503     }%
1504   }%
1505   {%
1506     \csedef{\@glxtr@csname}%
1507     {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
1508   }%
1509 }%
1510 {}%
1511 }

```

r@currunitcount

```

1512 \newcommand*{\@glxtr@currunitcount}[2]{%
1513   \ifcsundef
1514   {glo@\glsdetoklabel{#1}@currunit@#2}%
1515   {0}%
1516   {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}%
1517 }%

```

r@prevunitcount

```

1518 \newcommand*{\@glxtr@prevunitcount}[2]{%
1519   \ifcsundef
1520   {glo@\glsdetoklabel{#1}@prevunit@#2}%
1521   {0}%
1522   {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}%
1523 }%

```

eentryunitcount

```

1524 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
1525   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%

```

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

```

1526 \renewcommand*{\gls@defdocnewglossaryentry}{%
1527   \renewcommand*\newglossaryentry[2]{%
1528     \PackageError{glossaries}{\string\newglossaryentry\space
1529       may only be used in the preamble when entry counting has
1530       been activated}{If you use \string\glsenableentryunitcount\space
1531       you must place all entry definitions in the preamble not in
1532       the document environment}}%
1533   }%
1534 }%
```

New commands to access new fields:

```

1535 \newcommand*\glsentrycurrcount[1]{%
1536   \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
1537   \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
1538 }%
1539 \newcommand*\glsentryprevcount[1]{%
1540   \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
1541   \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
1542 }%
```

Access total count:

```

1543 \newcommand*\glsentryprevtotalcount[1]{%
1544   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
1545     {0}%
1546     {%
1547       \number\csuse{glo@\glsdetoklabel{##1}@prevunittotal}
1548     }%
1549 }%
```

Access max value:

```

1550 \newcommand*\glsentryprevmaxcount[1]{%
1551   \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
1552     {0}%
1553     {%
1554       \number\csuse{glo@\glsdetoklabel{##1}@prevunitmax}
1555     }%
1556 }%
```

Adjust post unset and reset:

```

1557 \let\@glsxtr@entryunitcount@org@unset\glsxtrpostunset
1558 \renewcommand*\glsxtrpostunset[1]{%
1559   \@glsxtr@entryunitcount@org@unset{##1}%
1560   \@gls@increment@currunitcount{##1}%
1561 }%
1562 \let\@glsxtr@entryunitcount@org@localunset\glsxtrpostlocalunset
1563 \renewcommand*\glsxtrpostlocalunset[1]{%
1564   \@glsxtr@entryunitcount@org@localunset{##1}%
1565   \@gls@local@increment@currunitcount{##1}%
1566 }%
1567 \let\@glsxtr@entryunitcount@org@reset\glsxtrpostreset
```



```

1568 \renewcommand*{\glstrpostreset}[1]{%
1569   \glshasattribute{##1}{unitcount}%
1570   {%
1571     \edef\@glstr@csname{\@glstr@currentunitcount{##1}}%
1572     \ifcsundef{\@glstr@csname}%
1573     {}%
1574     {\csgdef{\@glstr@csname}{0}}%
1575   }%
1576   {}%
1577 }%
1578 \let\@glstr@entryunitcount@org@localreset\glstrpostlocalreset
1579 \renewcommand*{\glstrpostlocalreset}[1]{%
1580   \@glstr@entryunitcount@org@localreset{##1}%
1581   \glshasattribute{##1}{unitcount}%
1582   {%
1583     \edef\@glstr@csname{\@glstr@currentunitcount{##1}}%
1584     \ifcsundef{\@glstr@csname}%
1585     {}%
1586     {\csdef{\@glstr@csname}{0}}%
1587   }%
1588   {}%
1589 }%

```

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

```

1590 \let\@cgl@@cgl@
1591 \let\@cgl@pl@@cgl@pl@
1592 \let\@cGL@@cGL@
1593 \let\@cGL@pl@@cGL@pl@
1594 \let\@cGL@@cGL@
1595 \let\@cGL@pl@@cGL@pl@

```

Write information to the aux file.

```

1596 \AtEndDocument{\@gls@write@entryunitcounts}%
1597 \renewcommand*{\@gls@entry@unitcount}[3]{%
1598   \csgdef{glo@glsdetoklabel{##1}@prevunit@##3}{##2}%
1599   \ifcsundef{glo@glsdetoklabel{##1}@prevunittotal}%
1600   {\csgdef{glo@glsdetoklabel{##1}@prevunittotal}{##2}}%
1601   {%
1602     \csxdef{glo@glsdetoklabel{##1}@prevunittotal}{
1603       \number\numexpr\csuse{glo@glsdetoklabel{##1}@prevunittotal}+##2}%
1604     }%
1605     \ifcsundef{glo@glsdetoklabel{##1}@prevunitmax}%
1606     {\csgdef{glo@glsdetoklabel{##1}@prevunitmax}{##2}}%
1607     {%
1608       \ifnum\csuse{glo@glsdetoklabel{##1}@prevunitmax}<##2
1609       \csgdef{glo@glsdetoklabel{##1}@prevunitmax}{##2}%
1610       \fi
1611     }%
1612 }%

```

```

1613 \let\glsenableentryunitcount\relax
1614 \renewcommand*{\glsenableentrycount}{%
1615   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
1616     can't be used with \string\glsenableentryunitcount}%
1617   {Use one or other but not both commands}%
1618 }%
1619 }
1620 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

1621 \newcommand*{\@gls@entry@unitcount}[3]{%

```

ryunitcounts@do

```

1622 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
1623   \immediate\write\@auxout
1624   {\string\@gls@entry@unitcount
1625     {\@glsentry}%
1626     {\@glsxtr@currunitcount{\@glsentry}{#1}%
1627     }%
1628     {#1}}%
1629 }

```

entryunitcounts

```

1630 \newcommand*{\@gls@write@entryunitcounts}{%
1631   \immediate\write\@auxout
1632   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
1633   \count@=0\relax
1634   \forallglsentries{\@glsentry}{%
1635     \gls@hasattribute{\@glsentry}{unitcount}%
1636     {%
1637       \ifglsused{\@glsentry}%
1638       {%
1639         \forlistcsloop
1640           {\@gls@write@entryunitcounts@do}%
1641           {glo@\glsdetoklabel{\@glsentry}@unitlist}%
1642       }%
1643     }%
1644     \advance\count@ by \@ne
1645   }%
1646 }%
1647 }%
1648 \ifnum\count@=0
1649   \GlossariesExtraWarningNoLine{Entry counting has been enabled
1650     \MessageBreak with \string\glsenableentryunitcount\space but the
1651     \MessageBreak attribute 'unitcount' hasn't
1652     \MessageBreak been assigned to any of the defined
1653     \MessageBreak entries}%
1654 \fi
1655 }

```

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.

```
1656 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
```

Enable entry counting:

```
1657 \glsenableentryunitcount
```

Redefine \gls etc:

```
1658 \renewcommand*{\gls}{\cglsl}%
1659 \renewcommand*{\Gls}{\cGls}%
1660 \renewcommand*{\glspl}{\cglspl}%
1661 \renewcommand*{\Glspl}{\cGlspl}%
1662 \renewcommand*{\GLS}{\cGLS}%
1663 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
1664 \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%
```

In case this command is used again:

```
1665 \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
1666 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
1667 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
1668 can't be used with \string\GlsXtrEnableEntryUnitCounting}%
1669 {Use one or other but not both commands}}%
1670 }
```

countunsetattr

```
1671 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
1672 \@for\@glsxtr@cat:=#1\do
1673 {%
1674 \ifdefempty{\@glsxtr@cat}{}%
1675 {%
1676 \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
1677 \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
1678 }%
1679 }%
1680 }
```

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

```
1681 \renewcommand*{\SetGenericNewAcronym}{%
```

```

1682 \let\@Gls@entryname\@Gls@acentryname
1683 \renewcommand{\newacronym}[4][\%
1684   \ifdefempty{\@glsacronymlists}%
1685   {%
1686     \def\@glo@type{\acronymtype}%
1687     \setkeys{glossentry}{##1}%
1688     \DeclareAcronymList{\@glo@type}%
1689   }%
1690   {%
1691     \glskeylisttok{##1}%
1692     \glslabeltok{##2}%
1693     \glsshorttok{##3}%
1694     \glslongtok{##4}%
1695     \newacronymhook
1696     \protected@edef\@do@newglossaryentry{%
1697       \noexpand\newglossaryentry{\the\glslabeltok}%
1698       {%
1699         type=\acronymtype,%
1700         name={\expandonce{\acronymentry{##2}}},%
1701         sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
1702         text={\the\glsshorttok},%
1703         short={\the\glsshorttok},%
1704         shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
1705         long={\the\glslongtok},%
1706         longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
1707         category=acronym,%
1708         \GenericAcronymFields,%
1709         \the\glskeylisttok
1710       }%
1711     }%
1712     \@do@newglossaryentry
1713   }%
1714 \renewcommand*{\acrfullfmt}[3]{%
1715   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
1716 \renewcommand*{\Acrfullfmt}[3]{%
1717   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
1718 \renewcommand*{\ACRfullfmt}[3]{%
1719   \glslink[##1]{##2}{%
1720     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
1721 \renewcommand*{\acrfullplfmt}[3]{%
1722   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
1723 \renewcommand*{\Acrfullplfmt}[3]{%
1724   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
1725 \renewcommand*{\ACRfullplfmt}[3]{%
1726   \glslink[##1]{##2}{%
1727     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
1728 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
1729 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
1730 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%

```

```

1731 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
1732 }

```

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:

```

1733 \let\@glxtr@org@setacronymstyle\setacronymstyle
1734 \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`.

```

1735 \newcommand*{\MakeAcronymsAbbreviations}{%
1736   \renewcommand*{\newacronym}[4][]{%
1737     \newabbreviation[type=\acronymtype,category=acronym,##1]{##2}{##3}{##4}%
1738   }%
1739   \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
1740   \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%
1741   \renewcommand*{\setacronymstyle}[1]{%
1742     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}
1743       unavailable.
1744       Use \string\setabbreviationstyle\space instead.
1745       The original acronym interface can be restored with
1746       \string\RestoreAcronyms}{}%
1747   }%
1748   \renewcommand*{\newacronymstyle}[1]{%
1749     \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
1750       available unless you restore the original acronym interface with
1751       \string\RestoreAcronyms}%
1752     \@glxtr@org@newacronymstyle{##1}%
1753   }%
1754 }

```

Switch acronyms to abbreviations:

```

1755 \MakeAcronymsAbbreviations

```

RestoreAcronyms Restore acronyms to glossaries interface.

```

1756 \newcommand*{\RestoreAcronyms}{%
1757   \SetGenericNewAcronym
1758   \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
1759   \renewcommand{\acronymfont}[1]{##1}%
1760   \let\setacronymstyle\@glxtr@org@setacronymstyle
1761   \let\newacronymstyle\@glxtr@org@newacronymstyle
1762   \let\@glsl@link@checkfirsthyper\@glxtr@org@checkfirsthyper
1763   \glsssetcategoryattribute{acronym}{regular}{false}%
1764   \setacronymstyle{long-short}%
1765 }

```

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

```

1766 \renewcommand*{\glsacspace}[1]{%
1767   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
1768   \ifdim\dimen@<\glsacspacemax~\else\space\fi
1769 }
```

`\glsacspacemax` Value used in the above.

```

1770 \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`

```

1771 \newcommand*{\@glxtr@reg@glosslist}{}
```

Save the original definition of `\makeglossaries`:

```

1772 \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`

```

1773 \renewcommand*{\makeglossaries}[1] [] {%
1774   \ifblank{#1}%
1775   {\@glxtr@org@makeglossaries}%
1776   {%
1777     \edef\@glxtr@reg@glosslist{#1}%
1778     \ifundef{\glswrite}{\newwrite\glswrite}{}%
1779     \protected@write\@auxout{}\string\providecommand
1780       \string\@glorder[1]{}}
1781     \protected@write\@auxout{}\string\providecommand
1782       \string\@istfilename[1]{}}
1783     \protected@write\@auxout{}\string\@istfilename{\istfilename}}%
1784     \protected@write\@auxout{}\string\@glorder{\glorder}}
1785     \write\@auxout{\string\providecommand\string\@gl@reference[3]{}}%
```

Iterate through each supplied glossary type and activate it.

```

1786   \@for\@glo@type:=#1\do{%
1787     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
1788   }%
```

New glossaries must be created before `\makeglossaries`:

```

1789   \renewcommand*{\newglossary}[4] [] {%
1790   \PackageError{glossaries}{New glossaries
```

```

1791 must be created before \string\makeglossaries}{You need
1792 to move \string\makeglossaries\space after all your
1793 \string\newglossary\space commands}}%

```

Any subsequence instances of this command should have no effect

```

1794 \let\@makeglossary\relax
1795 \let\makeglossary\relax
1796 \let\makeglossaries\relax

```

Disable all commands that have no effect after \makeglossaries

```

1797 \@disable@onlypremakeg

```

Allow see key:

```

1798 \let\gls@checkseeallowed\relax

```

Suppress warning about no \makeglossaries

```

1799 \let\warn@nomakeglossaries\relax
1800 \def\warn@noprintglossary{%
1801   \GlossariesWarningNoLine{No \string\printglossary\space
1802     or \string\printglossaries\space
1803     found.^^J(Remove \string\makeglossaries\space if you don't
1804 want
1805     any glossaries.)^^JThis document will not have a glossary}%
1806 }%

```

Adjust display number list to check for type:

```

1807 \renewcommand*\glsdisplaynumberlist}[1]{%
1808   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
1809   {\@glsxtr@idx@displaynumberlist{##1}}%
1810   {\@glsxtr@noidx@displaynumberlist{##1}}%
1811 }%

```

Adjust entry list:

```

1812 \renewcommand*\glsentrynumberlist}[1]{%
1813   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
1814   {\@glsxtr@idx@entrynumberlist{##1}}%
1815   {\@glsxtr@noidx@entrynumberlist{##1}}%
1816 }%

```

Adjust number list loop

```

1817 \renewcommand*\glsnumberlistloop}[2]{%
1818   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
1819   {%
1820     \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
1821       not available for glossary '##1'}{}%
1822   }%
1823   {\@glsxtr@noidx@numberlistloop{##1}{##2}}%
1824 }%

```

Only sanitize sort for normal indexing glossaries.

```

1825 \renewcommand*\glsprestandardsort}[3]{%
1826   \expandafter\DTLifinlist\expandafter{##2}{\@glsxtr@reg@glosslist}%

```

```

1827   {%
1828     \glsdosanitizesort
1829   }%
1830   {%
1831     \ifglssanitizesort
1832       \@gls@noidx@sanitizesort
1833     \else
1834       \@gls@noidx@nosanitizesort
1835     \fi
1836   }%
1837 }%

```

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

```

1838 \renewcommand*\new@glossaryentry[2]{%
1839   \PackageError{glossaries-extra}{Glossary entries must be defined
1840     in the preamble\MessageBreak when you use the optional argument
1841     of \string\makeglossaries}{Either move your definitions to the
1842     preamble or don't use the optional argument of
1843     \string\makeglossaries}%
1844 }%

```

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).

```

1845 \renewcommand*\@printgloss@setsort{%
1846   \renewcommand*\@glo@assign@sortkey{%
1847     \edef\@glo@type{\@glo@type}%
1848     \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glsxtr@reg@glosslist}%
1849     {%
1850       \@glo@no@assign@sortkey
1851     }%
1852     {%
1853       \@glo@assign@sortkey
1854     }%
1855   }%
1856   \def\@glo@sorttype{\@glo@default@sorttype}%
1857 }%

```

Check automake setting:

```

1858 \ifglsautomake
1859   \renewcommand*\@gls@doautomake{%
1860     \@for\@gls@type:=\@glsxtr@reg@glosslist\do{%
1861       \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
1862     }%
1863   }%
1864 \fi
1865 }%
1866 }

```

Display number list for the regular version:

splaynumberlist

```
1867 \let\@glstr@idx@displaynumberlist\glstrdisplaynumberlist
```

Display number list for the “noidx” version:

splaynumberlist

```
1868 \newcommand*{\@glstr@noidx@displaynumberlist}[1]{%
1869   \letcs{\@glstr@loclist}{\glsdetoklabel{#1}@loclist}%
1870   \ifdef\@glstr@loclist
1871   {%
1872     \def\@glstr@noidx@loclist@sep{%
1873       \def\@glstr@noidx@loclist@sep{%
1874         \def\@glstr@noidx@loclist@sep{%
1875           \glsnumlistsep
1876         }%
1877         \def\@glstr@noidx@loclist@finalsep{\glsnumlistlastsep}%
1878       }%
1879     }%
1880     \def\@glstr@noidx@loclist@finalsep{}%
1881     \def\@glstr@noidx@loclist@prev{}%
1882     \forlistloop{\glsnoidxdisplayloclisthandler}{\@glstr@loclist}%
1883     \@glstr@noidx@loclist@finalsep
1884     \@glstr@noidx@loclist@prev
1885   }%
1886   {%
1887     ??\glsdoifexists{#1}%
1888     {%
1889       \GlossariesWarning{Missing location list for ‘#1’. Either
1890         a rerun is required or you haven’t referenced the entry.}%
1891     }%
1892   }%
1893 }%
1894
```

And for the number list loop:

@numberlistloop

```
1895 \newcommand*{\@glstr@noidx@numberlistloop}[3]{%
1896   \letcs{\@glstr@loclist}{\glsdetoklabel{#1}@loclist}%
1897   \let\@glstr@org@glstrnoidxdisplayloc\glstrnoidxdisplayloc
1898   \let\@glstr@org@glstrseeformat\glstrseeformat
1899   \let\@glstrnoidxdisplayloc#2\relax
1900   \let\@glstrseeformat#3\relax
1901   \ifdef\@glstr@loclist
1902   {%
1903     \forlistloop{\glsnoidxnumberlistloophandler}{\@glstr@loclist}%
1904   }%
1905   {%
1906     ??\glsdoifexists{#1}%
1907     {%
```

```

1908      \GlossariesWarning{Missing location list for ‘##1’. Either
1909      a rerun is required or you haven’t referenced the entry.}%
1910  }%
1911 }%
1912 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
1913 \let\glsseeformat\@gls@org@glsseeformat
1914 }%

```

Same for entry number list.

entrynumberlist

```

1915 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
1916   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
1917   \ifdef\@gls@loclist
1918   {%
1919     \glsnoidxloclist{\@gls@loclist}%
1920   }%
1921   {%
1922     ??\glsdoifexists{#1}%
1923     {%
1924       \GlossariesWarning{Missing location list for ‘#1’. Either
1925       a rerun is required or you haven’t referenced the entry.}%
1926     }%
1927   }%
1928 }%

```

entrynumberlist

```

1929 \let\@glsxtr@idx@entrynumberlist\glsentrynumberlist

```

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

@print@glossary

```

1930 \renewcommand{\@print@glossary}{%
1931   \makeatletter
1932   \@input{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
1933   \IfFileExists{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
1934   {}%
1935   {\glsxtrNoGlossaryWarning{\@glo@type}}%
1936   \ifglsxindy
1937     \ifcsundef{@xdy@\@glo@type @language}%
1938     {%
1939       \edef\@do@auxoutstuff{%
1940         \noexpand\AtEndDocument{%
1941           \noexpand\immediate\noexpand\write\@auxout{%
1942             \string\providecommand\string\@xdylanguage[2]{}}%
1943           \noexpand\immediate\noexpand\write\@auxout{%
1944             \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
1945           }%
1946         }%

```

```

1947 }%
1948 {%
1949 \edef\@do@auxoutstuff{%
1950 \noexpand\AtEndDocument{%
1951 \noexpand\immediate\noexpand\write\@auxout{%
1952 \string\providecommand\string\@xdylanguage[2]{}}%
1953 \noexpand\immediate\noexpand\write\@auxout{%
1954 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
1955 @language\endcsname}}%
1956 }%
1957 }%
1958 }%
1959 \@do@auxoutstuff
1960 \edef\@do@auxoutstuff{%
1961 \noexpand\AtEndDocument{%
1962 \noexpand\immediate\noexpand\write\@auxout{%
1963 \string\providecommand\string\@gls@codepage[2]{}}%
1964 \noexpand\immediate\noexpand\write\@auxout{%
1965 \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
1966 }%
1967 }%
1968 \@do@auxoutstuff
1969 \fi
1970 \renewcommand*{\@warn@nomakeglossaries}{%
1971 \GlossariesWarningNoLine{\string\makeglossaries\space
1972 hasn't been used,^^Jthe glossaries will not be updated}%
1973 }%
1974 }

```

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

`\GlsWarningHead` Header message.

```

1975 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
1976 This document is incomplete. The external file associated with
1977 the glossary '#1' (which should be called \texttt{#2})
1978 hasn't been created.%
1979 }

```

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

```

1980 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
1981 This has probably happened because there are no entries defined
1982 in this glossary.%
1983 }

```

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

```

1984 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
1985 If you don't want this glossary,
1986 add \texttt{nomain} to your package option list when you load
1987 \texttt{glossaries-extra.sty}. For example:%
1988 }

```

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

```
1989 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
1990 Did you forget to use \texttt{type=#1} when you defined your
1991 entries? If you tried to load entries into this glossary with
1992 \texttt{\string\loadglsentries} did you remember to use
1993 \texttt{[#1]} as the optional argument? If you did, check that
1994 the definitions in the file you loaded all had the type set
1995 to \texttt{\string\glsdefaulttype}.\%
1996 }
```

arningCheckFile Advisory message to check the file contents.

```
1997 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
1998 Check the contents of the file \texttt{#1}. If
1999 it's empty, that means you haven't indexed any of your entries in this
2000 glossary (using commands like \texttt{\string\gls} or
2001 \texttt{\string\glsadd}) so this list can't be generated.
2002 If the file isn't empty, the document build process hasn't been
2003 completed.\%
2004 }
```

WarningAutoMake Message when automake option has been used.

```
2005 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
2006 You may need to rerun \LaTeX. If you already have, it may be that
2007 \TeX's shell escape doesn't allow you to run
2008 \ifglxindy xindy\else makeindex\fi. Check the
2009 transcript file \texttt{\jobname.log}. If the shell escape is
2010 disabled, try one of the following:
2011
2012 \begin{itemize}
2013 \item Run the external (Lua) application:
2014
2015 \texttt{makeglossaries-lite \string\jobname\string}
2016
2017 \item Run the external (Perl) application:
2018
2019 \texttt{makeglossaries \string\jobname\string}
2020 \end{itemize}
2021
2022 Then rerun \LaTeX\ on this document.
2023 \GlossariesExtraWarning{Rerun required to build the
2024 glossary '#1' or check TeX's shell escape allows
2025 you to run \ifglxindy xindy\else makeindex\fi}%
2026 }
```

WarningMisMatch Mismatching \makenoidxglossaries.

```
2027 \newcommand{\GlsXtrNoGlsWarningMisMatch}{%
2028 You need to either replace \texttt{\string\makenoidxglossaries}
2029 with \texttt{\string\makeglossaries} or replace
```

```

2030 \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
2031 \texttt{\string\printnoidxglossary}
2032 (or \texttt{\string\printnoidxglossaries}) and then rebuild
2033 this document.%
2034 }

```

arningBuildInfo Build advice.

```

2035 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
2036   Try one of the following:
2037   \begin{itemize}
2038     \item Add \texttt{automake} to your package option list when you load
2039       \texttt{glossaries-extra.sty}. For example:
2040
2041       \texttt{\string\usepackage[automake]%
2042         \glsopenbrace glossaries-extra\glsclosebrace}
2043
2044     \item Run the external (Lua) application:
2045
2046       \texttt{makeglossaries-lite \string"\jobname\string"}
2047
2048     \item Run the external (Perl) application:
2049
2050       \texttt{makeglossaries \string"\jobname\string"}
2051   \end{itemize}
2052
2053   Then rerun \LaTeX\ on this document.%
2054 }

```

oGlsWarningTail Final paragraph.

```

2055 \newcommand{\GlsXtrNoGlsWarningTail}{%
2056   This message will be removed once the problem has been fixed.%
2057 }

```

GlsWarningNoOut No out file created. Build advice.

```

2058 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
2059   The file \texttt{#1} doesn't exist. This most likely means you haven't used
2060   \texttt{\string\makeglossaries} or you have used
2061   \texttt{\string\nofiles}. If this is just a draft version of the
2062   document, you can suppress this message using the
2063   \texttt{nomissingglstext} package option.%
2064 }

```

glossarywarning

```

2065 \newcommand*{@@glxtr@defaultnoglossarywarning}[1]{%
2066   \glossarysection[\glossarytoctitle]{\glossarytitle}
2067   \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
2068   \par
2069   \glxtrifemptyglossary{#1}%
2070   {%

```

```

2071 \GlsXtrNoGlsWarningEmptyStart\space
2072 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
2073 \medskip
2074 \noindent\texttt{\string\usepackage[nomain\ifglsacronym ,acronym\fi]%
2075 \glsopenbrace glossaries-extra\glsclosebrace}
2076 \medskip
2077 }%
2078 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
2079 }%
2080 {%
2081 \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
2082 {%
2083 \GlsXtrNoGlsWarningCheckFile
2084 {\jobname.\csname @glotype@\@glo@type @out\endcsname}
2085
2086 \ifglsautomake
2087
2088 \GlsXtrNoGlsWarningAutoMake{#1}
2089
2090 \else
2091
2092 \ifthenelse{\equal{#1}{main}}%
2093 {%
2094 \GlsXtrNoGlsWarningEmptyMain\par
2095 \medskip
2096 \noindent\texttt{\string\usepackage[nomain]%
2097 \glsopenbrace glossaries-extra\glsclosebrace}
2098 \medskip
2099 }%
2100 {}%
2101
2102 \ifdefequal\makeglossaries\@no@makeglossaries
2103 {%
2104 \GlsXtrNoGlsWarningMisMatch
2105 }%
2106 {%
2107 \GlsXtrNoGlsWarningBuildInfo
2108 }%
2109 \fi
2110 }%
2111 {%
2112 \GlsXtrNoGlsWarningNoOut
2113 {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
2114 }%
2115 }%
2116 \par
2117 \GlsXtrNoGlsWarningTail
2118 }

```

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.

```
2119 \@ifpackageloaded{glossaries-accsupp}
2120 {
```

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

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

```
2121 \newcommand*{\glsaccessname}[1]{%
2122 \glsnameaccessdisplay
2123 {%
2124 \glsentryname{#1}%
2125 }%
2126 {#1}%
2127 }
```

`\@glsname@` Redefine to use accessibility support.

```
2128 \def\@glsname@#1#2[#3]{%
2129 \@gls@field@link{#1}{#2}{\glsaccessname{#2}#3}%
2130 }
```

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

```
2131 \newcommand*{\Glsaccessname}[1]{%
2132 \glsnameaccessdisplay
2133 {%
2134 \Glsentryname{#1}%
2135 }%
2136 {#1}%
2137 }
```

`\@Glsname@` Redefine to use accessibility support.

```
2138 \def\@Glsname@#1#2[#3]{%
2139 \@gls@field@link{#1}{#2}{\Glsaccessname{#2}#3}%
2140 }
```

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

```
2141 \newcommand*{\GLSaccessname}[1]{%
2142 \glsnameaccessdisplay
2143 {%
2144 \mfirstucMakeUppercase{\glsentryname{#1}}%
2145 }%
2146 {#1}%
2147 }
```

\@GLSname@ Redefine to use accessibility support.

```
2148 \def\@GLSname@#1#2[#3]{%
2149   \@gls@field@link{#1}{#2}{\GLSaccessname{#2}#3}%
2150 }
```

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

```
2151   \newcommand*{\glsaccesstext}[1]{%
2152     \glstextaccessdisplay
2153     {%
2154       \glentrytext{#1}%
2155     }%
2156     {#1}%
2157   }
```

\@glstext@ Redefine to use accessibility support.

```
2158 \def\@glstext@#1#2[#3]{%
2159   \@gls@field@link{#1}{#2}{\glsaccesstext{#2}#3}%
2160 }
```

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

```
2161   \newcommand*{\Glsaccesstext}[1]{%
2162     \glstextaccessdisplay
2163     {%
2164       \Glentrytext{#1}%
2165     }%
2166     {#1}%
2167   }
```

\@Glstext@ Redefine to use accessibility support.

```
2168 \def\@Glstext@#1#2[#3]{%
2169   \@gls@field@link{#1}{#2}{\Glsaccesstext{#2}#3}%
2170 }
```

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

```
2171   \newcommand*{\GLSaccesstext}[1]{%
2172     \glstextaccessdisplay
2173     {%
2174       \mfirstucMakeUppercase{\glentrytext{#1}}%
2175     }%
2176     {#1}%
2177   }
```

\@GLStext@ Redefine to use accessibility support.

```
2178 \def\@GLStext@#1#2[#3]{%
2179   \@gls@field@link{#1}{#2}{\GLSaccesstext{#2}#3}%
2180 }
```


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

```
2181 \newcommand*{\glsaccessplural}[1]{%
2182   \glspluralaccessdisplay
2183   {%
2184     \glstryplural{#1}%
2185   }%
2186   {#1}%
2187 }
```

\@glsplural@ Redefine to use accessibility support.

```
2188 \def\@glsplural@#1#2[#3]{%
2189   \@gls@field@link{#1}{#2}{\glsaccessplural{#2}#3}%
2190 }
```

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

```
2191 \newcommand*{\Glsaccessplural}[1]{%
2192   \glspluralaccessdisplay
2193   {%
2194     \Glstryplural{#1}%
2195   }%
2196   {#1}%
2197 }
```

\@Glsplural@ Redefine to use accessibility support.

```
2198 \def\@Glsplural@#1#2[#3]{%
2199   \@gls@field@link{#1}{#2}{\Glsaccessplural{#2}#3}%
2200 }
```

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

```
2201 \newcommand*{\GLSaccessplural}[1]{%
2202   \glspluralaccessdisplay
2203   {%
2204     \mfirstucMakeUppercase{\glstryplural{#1}}%
2205   }%
2206   {#1}%
2207 }
```

\@GLSplural@ Redefine to use accessibility support.

```
2208 \def\@GLSplural@#1#2[#3]{%
2209   \@gls@field@link{#1}{#2}{\GLSaccessplural{#2}#3}%
2210 }
```

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

```
2211 \newcommand*{\glsaccessfirst}[1]{%
2212   \glsfirstaccessdisplay
2213   {%
2214     \glstryfirst{#1}%
2215   }%
2216   {#1}%
2217 }
```

```

2215     }%
2216     {#1}%
2217 }

```

`\@glsfirst@` Redefine to use accessibility support.

```

2218 \def\@glsfirst@#1#2[#3]{%
2219   \@gls@field@link{#1}{#2}{\glsaccessfirst{#2}#3}%
2220 }

```

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

```

2221 \newcommand*\Glsaccessfirst}[1]{%
2222   \glsfirstaccessdisplay
2223   {%
2224     \glentryfirst{#1}%
2225   }%
2226   {#1}%
2227 }

```

`\@Glsfirst@` Redefine to use accessibility support.

```

2228 \def\@Glsfirst@#1#2[#3]{%
2229   \@gls@field@link{#1}{#2}{\Glsaccessfirst{#2}#3}%
2230 }

```

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

```

2231 \newcommand*\GLSaccessfirst}[1]{%
2232   \glsfirstaccessdisplay
2233   {%
2234     \mfirstucMakeUppercase{\glentryfirst{#1}}%
2235   }%
2236   {#1}%
2237 }

```

`\@GLSfirst@` Redefine to use accessibility support.

```

2238 \def\@GLSfirst@#1#2[#3]{%
2239   \@gls@field@link{#1}{#2}{\GLSaccessfirst{#2}#3}%
2240 }

```

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

```

2241 \newcommand*\glsaccessfirstplural}[1]{%
2242   \glsfirstpluralaccessdisplay
2243   {%
2244     \glentryfirstplural{#1}%
2245   }%
2246   {#1}%
2247 }

```

```

glsfirstplural@  Redefine to use accessibility support.
2248 \def\@glsfirstplural@#1#2[#3]{%
2249   \@gls@field@link{#1}{#2}{\glsaccessfirstplural{#2}#3}%
2250 }

cessfirstplural  Display the firstplural value (no link and no check for existence) with the first letter converted
to upper case.
2251   \newcommand*{\Glsaccessfirstplural}[1]{%
2252     \glsfirstpluralaccessdisplay
2253     {%
2254       \Glsentryfirstplural{#1}%
2255     }%
2256     {#1}%
2257   }

Glsfirstplural@  Redefine to use accessibility support.
2258 \def\@Glsfirstplural@#1#2[#3]{%
2259   \@gls@field@link{#1}{#2}{\Glsaccessfirstplural{#2}#3}%
2260 }

cessfirstplural  Display the firstplural value (no link and no check for existence) converted to upper case.
2261   \newcommand*{\GLSaccessfirstplural}[1]{%
2262     \glsfirstpluralaccessdisplay
2263     {%
2264       \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
2265     }%
2266     {#1}%
2267   }

GLSfirstplural@  Redefine to use accessibility support.
2268 \def\@GLSfirstplural@#1#2[#3]{%
2269   \@gls@field@link{#1}{#2}{\GLSaccessfirstplural{#2}#3}%
2270 }

glsaccesssymbol  Display the symbol value (no link and no check for existence).
2271   \newcommand*{\glsaccesssymbol}[1]{%
2272     \glssymbolaccessdisplay
2273     {%
2274       \glsentrysymbol{#1}%
2275     }%
2276     {#1}%
2277   }

\@glssymbol@  Redefine to use accessibility support.
2278 \def\@glssymbol@#1#2[#3]{%
2279   \@gls@field@link{#1}{#2}{\glsaccesssymbol{#2}#3}%
2280 }

```

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

```

2281 \newcommand*\Glsaccesssymbol}[1]{%
2282   \glssymbolaccessdisplay
2283   {%
2284     \glentrysymbol{#1}%
2285   }%
2286   {#1}%
2287 }
```

\@Glsymbol@ Redefine to use accessibility support.

```

2288 \def\@Glsymbol@#1#2[#3]{%
2289   \@gls@field@link{#1}{#2}{\Glsaccesssymbol{#2}#3}%
2290 }
```

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

```

2291 \newcommand*\GLSaccesssymbol}[1]{%
2292   \glssymbolaccessdisplay
2293   {%
2294     \mfirstucMakeUppercase{\glentrysymbol{#1}}%
2295   }%
2296   {#1}%
2297 }
```

\@GLSsymbol@ Redefine to use accessibility support.

```

2298 \def\@GLSsymbol@#1#2[#3]{%
2299   \@gls@field@link{#1}{#2}{\GLSaccesssymbol{#2}#3}%
2300 }
```

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

```

2301 \newcommand*\glsaccesssymbolplural}[1]{%
2302   \glssymbolpluralaccessdisplay
2303   {%
2304     \glentrysymbolplural{#1}%
2305   }%
2306   {#1}%
2307 }
```

lssymbolplural@ Redefine to use accessibility support.

```

2308 \def\@glssymbolplural@#1#2[#3]{%
2309   \@gls@field@link{#1}{#2}{\glsaccesssymbolplural{#2}#3}%
2310 }
```

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

```

2311 \newcommand*\Glsaccesssymbolplural}[1]{%
2312   \glssymbolpluralaccessdisplay
2313   {%
```

```

2314     \Glsentrysymbolplural{#1}%
2315 }%
2316 {#1}%
2317 }

```

lssymbolplural@ Redefine to use accessibility support.

```

2318 \def\@Glsymbolplural@#1#2[#3]{%
2319   \@gls@field@link{#1}{#2}{\Glsaccesssymbolplural{#2}#3}%
2320 }

```

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

```

2321   \newcommand*{\Glsaccesssymbolplural}[1]{%
2322     \glssymbolpluralaccessdisplay
2323     {%
2324       \mfirstucMakeUppercase{\glsentrysymbolplural{#1}}%
2325     }%
2326     {#1}%
2327   }

```

LSsymbolplural@ Redefine to use accessibility support.

```

2328 \def\@GLSsymbolplural@#1#2[#3]{%
2329   \@gls@field@link{#1}{#2}{\GLSaccesssymbolplural{#2}#3}%
2330 }

```

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

```

2331   \newcommand*{\glsaccessdesc}[1]{%
2332     \glsdescriptionaccessdisplay
2333     {%
2334       \glsentrydesc{#1}%
2335     }%
2336     {#1}%
2337   }

```

\@glsdesc@ Redefine to use accessibility support.

```

2338 \def\@glsdesc@#1#2[#3]{%
2339   \@gls@field@link{#1}{#2}{\glsaccessdesc{#2}#3}%
2340 }

```

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

```

2341   \newcommand*{\Glsaccessdesc}[1]{%
2342     \glsdescriptionaccessdisplay
2343     {%
2344       \Glsentrydesc{#1}%
2345     }%
2346     {#1}%
2347   }

```

```

\@GLSdesc@   Redefine to use accessibility support.
2348 \def\@GLSdesc@#1#2[#3]{%
2349   \@gls@field@link{#1}{#2}{\GLSaccessdesc{#2}#3}%
2350 }

\GLSaccessdesc   Display the desc value (no link and no check for existence) converted to upper case.
2351   \newcommand*{\GLSaccessdesc}[1]{%
2352     \glsdescriptionaccessdisplay
2353     {%
2354       \mfirstucMakeUppercase{\glsentrydesc{#1}}%
2355     }%
2356     {#1}%
2357   }

\@GLSdesc@   Redefine to use accessibility support.
2358 \def\@GLSdesc@#1#2[#3]{%
2359   \@gls@field@link{#1}{#2}{\GLSaccessdesc{#2}#3}%
2360 }

accessdescplural   Display the descplural value (no link and no check for existence).
2361   \newcommand*{\glsaccessdescplural}[1]{%
2362     \glsdescriptionpluralaccessdisplay
2363     {%
2364       \glsentrydescplural{#1}%
2365     }%
2366     {#1}%
2367   }

@glsdescplural@   Redefine to use accessibility support.
2368 \def\@glsdescplural@#1#2[#3]{%
2369   \@gls@field@link{#1}{#2}{\glsaccessdescplural{#2}#3}%
2370 }

accessdescplural   Display the descplural value (no link and no check for existence) with the first letter converted
to upper case.
2371   \newcommand*{\Glsaccessdescplural}[1]{%
2372     \glsdescriptionpluralaccessdisplay
2373     {%
2374       \Glsentrydescplural{#1}%
2375     }%
2376     {#1}%
2377   }

@Glsdescplural@   Redefine to use accessibility support.
2378 \def\@Glsdescplural@#1#2[#3]{%
2379   \@gls@field@link{#1}{#2}{\Glsaccessdescplural{#2}#3}%
2380 }

```

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

```

2381 \newcommand*{\GLSaccessdescplural}[1]{%
2382   \glsdescriptionpluralaccessdisplay
2383   {%
2384     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
2385   }%
2386   {#1}%
2387 }
```

@GLSdescplural@ Redefine to use accessibility support.

```

2388 \def\@GLSdescplural@#1#2[#3]{%
2389   \@gls@field@link{#1}{#2}{\GLSaccessdescplural{#2}#3}%
2390 }
```

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

```

2391 \newcommand*{\glsaccessshort}[1]{%
2392   \glsshortaccessdisplay
2393   {%
2394     \glsentryshort{#1}%
2395   }%
2396   {#1}%
2397 }
```

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

```

2398 \newcommand*{\Glsaccessshort}[1]{%
2399   \glsshortaccessdisplay
2400   {%
2401     \Glsentryshort{#1}%
2402   }%
2403   {#1}%
2404 }
```

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

```

2405 \newcommand*{\GLSaccessshort}[1]{%
2406   \glsshortaccessdisplay
2407   {%
2408     \mfirstucMakeUppercase{\glsentryshort{#1}}%
2409   }%
2410   {#1}%
2411 }
```

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

```

2412 \newcommand*{\glsaccessshortpl}[1]{%
2413   \glsshortpluralaccessdisplay
2414   {%
2415     \glsentryshortpl{#1}%
2416   }%
```

```

2417     {#1}%
2418 }

```

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

```

2419 \newcommand*{\lsaccessshortpl}[1]{%
2420   \glsshortpluralaccessdisplay
2421   {%
2422     \Glentryshortpl{#1}%
2423   }%
2424   {#1}%
2425 }

```

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

```

2426 \newcommand*{\LSaccessshortpl}[1]{%
2427   \glsshortpluralaccessdisplay
2428   {%
2429     \mfirstucMakeUppercase{\glentryshortpl{#1}}%
2430   }%
2431   {#1}%
2432 }

```

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

```

2433 \newcommand*{\glsaccesslong}[1]{%
2434   \glslongaccessdisplay{\glentrylong{#1}}{#1}%
2435 }

```

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

```

2436
2437 \newcommand*{\Glsaccesslong}[1]{%
2438   \glslongaccessdisplay{\Glentrylong{#1}}{#1}%
2439 }

```

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

```

2440 \newcommand*{\GLSaccesslong}[1]{%
2441   \glslongaccessdisplay
2442   {%
2443     \mfirstucMakeUppercase{\glentrylong{#1}}%
2444   }%
2445   {#1}%
2446 }

```

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

```

2447 \newcommand*{\glsaccesslongpl}[1]{%
2448   \glslongpluralaccessdisplay{\glentrylongpl{#1}}{#1}%
2449 }

```


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

```
2450
2451 \newcommand*{\Glsaccesslongpl}[1]{%
2452   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
2453 }
```

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

```
2454 \newcommand*{\GLSaccesslongpl}[1]{%
2455   \glslongpluralaccessdisplay
2456   {%
2457     \mfirstucMakeUppercase{\Glsentrylongpl{#1}}%
2458   }%
2459   {#1}%
2460 }
```

End of if part

```
2461 }
2462 {
```

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

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

```
2463 \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.

```
2464 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}
```

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

```
2465 \newcommand*{\GLSAccessname}[1]{%
2466   \protect\mfirstucMakeUppercase{\glsentryname{#1}}}
```

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

```
2467 \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.

```
2468 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}
```

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

```
2469 \newcommand*{\GLSAccesstext}[1]{%
2470   \protect\mfirstucMakeUppercase{\glsentrytext{#1}}}
```

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

```
2471 \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.

```
2472 \newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}
```

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

```

2473 \newcommand*{\GLSaccessplural}[1]{%
2474 \protect\mfirstucMakeUppercase{\glentryplural{#1}}}
```

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

```

2475 \newcommand*{\glaccessfirst}[1]{\glentryfirst{#1}}
```

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

```

2476 \newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}
```

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

```

2477 \newcommand*{\GLSaccessfirst}[1]{%
2478 \protect\mfirstucMakeUppercase{\glentryfirst{#1}}}
```

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

```

2479 \newcommand*{\glaccessfirstplural}[1]{\glentryfirstplural{#1}}
```

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

```

2480 \newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}
```

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

```

2481 \newcommand*{\GLSaccessfirstplural}[1]{%
2482 \protect\mfirstucMakeUppercase{\glentryfirstplural{#1}}}
```

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

```

2483 \newcommand*{\glaccesssymbol}[1]{\glentrysymbol{#1}}
```

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

```

2484 \newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}
```

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

```

2485 \newcommand*{\GLSaccesssymbol}[1]{%
2486 \protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}
```

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

```

2487 \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.

```

2488 \newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}
```

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

```

2489 \newcommand*{\GLSaccesssymbolplural}[1]{%
2490 \protect\mfirstucMakeUppercase{\glentrysymbolplural{#1}}}
```

`\glsaccessdesc` Display the desc value (no link and no check for existence).
2491 `\newcommand*{\glsaccessdesc}[1]{\glsentrydesc{#1}}`

`\Glsaccessdesc` Display the desc value (no link and no check for existence) with the first letter converted to upper case.
2492 `\newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence). converted to upper case.
2493 `\newcommand*{\GLSaccessdesc}[1]{%`
2494 `\protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}`

`ccessdescplural` Display the descplural value (no link and no check for existence).
2495 `\newcommand*{\glsaccessdescplural}[1]{\glsentrydescplural{#1}}`

`ccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.
2496 `\newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}`

`ccessdescplural` Display the descplural value (no link and no check for existence). converted to upper case.
2497 `\newcommand*{\GLSaccessdescplural}[1]{%`
2498 `\protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}`

`\glsaccessshort` Display the short form (no link and no check for existence).
2499 `\newcommand*{\glsaccessshort}[1]{\glsentryshort{#1}}`

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).
2500 `\newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}`

`\GLSaccessshort` Display the short value (no link and no check for existence). converted to upper case.
2501 `\newcommand*{\GLSaccessshort}[1]{%`
2502 `\protect\mfirstucMakeUppercase{\glsentryshort{#1}}}`

`lsaccessshortpl` Display the short plural form (no link and no check for existence).
2503 `\newcommand*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}`

`lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).
2504 `\newcommand*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}`

`LSaccessshortpl` Display the shortplural value (no link and no check for existence). converted to upper case.
2505 `\newcommand*{\GLSaccessshortpl}[1]{%`
2506 `\protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}`

`\glsaccesslong` Display the long form (no link and no check for existence).
2507 `\newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}`

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

```
2508 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

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

```
2509 \newcommand*{\GLSaccesslong}[1]{%
2510 \protect\mfirstucMakeUppercase{\Glsentrylong{#1}}}
```

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

```
2511 \newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}
```

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

```
2512 \newcommand*{\GLSaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

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

```
2513 \newcommand*{\GLSaccesslongpl}[1]{%
2514 \protect\mfirstucMakeUppercase{\Glsentrylongpl{#1}}}
```

End of else part

```
2515 }
```

1.5 Categories

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

```
2516 \glsaddstoragekey{category}{general}{\glscategory}
```

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

```
2517 \newcommand{\glsifcategory}[4]{%
2518 \ifglsseldeq{#1}{category}{#2}{#3}{#4}%
2519 }
```

Categories can have attributes.

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

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

```
2520 \newcommand*{\glssetcategoryattribute}[3]{%
2521 \csdef{@glsextr@categoryattr@@#1@#2}{#3}%
2522 }
```

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

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

```
2523 \newcommand*{\glsgetcategoryattribute}[2]{%
2524   \csuse{@glsxtr@categoryattr@#1@#2}%
2525 }
```

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

Tests if the category has the given attribute set.

```
2526 \newcommand*{\glshascategoryattribute}[4]{%
2527   \ifcsvoid{@glsxtr@categoryattr@#1@#2}{#4}{#3}%
2528 }
```

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

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

```
2529 \newcommand*{\glsssetAttribute}[3]{%
2530   \glsssetcategoryattribute{\glscategory{#1}}{#2}{#3}%
2531 }
```

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

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

```
2532 \newcommand*{\glsgsetAttribute}[2]{%
2533   \glsgsetcategoryattribute{\glscategory{#1}}{#2}%
2534 }
```

\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.

```
2535 \newcommand*{\glshasattribute}[4]{%
2536   \ifglentryexists{#1}%
2537   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
2538   {#4}%
2539 }
```

categoryattribute

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

True if category has the attribute with the given value.

```
2540 \newcommand{\glsifcategoryattribute}[5]{%
2541   \ifcsundef{@glstr@categoryattr@#1@#2}%
2542   {#5}%
2543   {\ifcsstring{@glstr@categoryattr@#1@#2}{#3}{#4}{#5}}%
2544 }
```

\glsifattribute

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

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

```
2545 \newcommand{\glsifattribute}[5]{%
2546   \ifglentryexists{#1}%
2547   {\glsifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
2548   {#5}%
2549 }
```

Set attributes for the default general category:

```
2550 \glsetcategoryattribute{general}{regular}{true}
```

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

```
2551 \glsetcategoryattribute{acronym}{regular}{true}
```

regularcategory

Convenient shortcut to create add the regular attribute.

```
2552 \newcommand*{\glsetregularcategory}[1]{%
2553   \glsetcategoryattribute{#1}{regular}{true}%
2554 }
```

fregularcategory

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

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

```
2555 \newcommand{\glsifregularcategory}[3]{%
2556   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
2557 }
```

\glsifregular

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

Short cut to determine if an entry has a regular category.

```
2558 \newcommand{\glsifregular}[3]{%
2559   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
2560 }
```

oreachincategory

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

Iterates through all entries in all the glossaries (or just those listed in *glossary labels*) and does *body* if the category matches *category-label*. The control sequences *glossary-cs* and *label-cs* may be used in *body* to access the glossary label and entry label for the current iteration.

```
2561 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
2562   \forallglossaries[#1]{#3}%
2563   {%
2564     \forglentries[#3]{#4}%
2565     {%
2566       \glsifcategory{#4}{#2}{#5}{}%
2567     }%
2568   }%
2569 }
```

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.

```
2570 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
2571   \forallglossaries[#1]{#4}%
2572   {%
2573     \forglentries[#4]{#5}%
2574     {%
2575       \glsifattribute{#5}{#2}{#3}{#6}{}%
2576     }%
2577   }%
2578 }
```

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

```
2579 \ifdef\newterm
2580 {%
```

`\newterm`

```
2581 \renewcommand*{\newterm}[2] [] {%
2582   \newglossaryentry{#2}%
2583     {type={index},category=index,name={#2},%
2584       description={\glstrpostdescindex\nopostdesc},#1}%
2585 }
```

Indexed terms are regular by default.

```
2586 \glsetcategoryattribute{index}{regular}{true}
```

`trpostdescindex`

```
2587 \newcommand*{\glstrpostdescindex}{}
2588 }
2589 {}
```

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`.

```
2590 \ifdef\printsymbols
2591 {%
```

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

```
2592 \newcommand*{\glstrnewsymbol}[3] [] {%
2593   \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
2594 }
```

Symbols are regular by default.

```
2595 \glsetcategoryattribute{symbol}{regular}{true}
```

`rpostdescsymbol`

```
2596 \newcommand*{\glstrpostdescsymbol}{}
2597 }
2598 {}
```

Similar for the `numbers` option.

```
2599 \ifdef\printnumbers
2600 {%
```

`glstrnewnumber`

```
2601 \ifdef\printnumbers
2602   \newcommand*{\glstrnewnumber}[3] [] {%
2603     \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
2604   }
```

Numbers are regular by default.

```
2605 \glsetcategoryattribute{number}{regular}{true}
```


rpostdescnumber

```
2606 \newcommand*{\glxtrpostdescnumber}{}  
2607 }  
2608 {}
```

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.

```
2609 \newcommand*{\glxtrsetcategory}[2]{%  
2610 \for\@glxtr@label:=#1\do  
2611 {%  
2612 \glsfieldxddef{\@glxtr@label}{category}{#2}%  
2613 }%  
2614 }
```

tcategoryforall 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.

```
2615 \newcommand*{\glxtrsetcategoryforall}[2]{%  
2616 \forallglossaries[#1]{\@glxtr@type}{%  
2617 \for\glsentries[\@glxtr@type]{\@glxtr@label}%  
2618 {%  
2619 \glsfieldxddef{\@glxtr@label}{category}{#2}%  
2620 }%  
2621 }%  
2622 }
```

trfieldtitlecase

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

Apply title casing to the contents of the given field.

```
2623 \newcommand*{\glxtrfieldtitlecase}[2]{%  
2624 \expandafter\xcapitalisewords\expandafter  
2625 {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%  
2626 }
```

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.

```
2627 \@ifpackageloaded{glossaries-accsupp}  
2628 {  
2629 \renewcommand*{\glossentrydesc}[1]{%  
2630 \glsdoifexistsorwarn{#1}%  
2631 {%  
2632 \glssetabbrvfmt{\glscategory{#1}}%  
2633 \glsifattribute{#1}{glossdesc}{firstuc}%  
2634 }  
2635 }
```

```

2634      {%
2635      \Glsaccessdesc{#1}%
2636      }%
2637      {%
2638      \glsifattribute{#1}{glossdesc}{title}%
2639      {%
2640      \@glstr@do@titlecaps@warn
2641      \glsdescriptionaccessdisplay
2642      {%
2643      \glstrfieldtitlecase{#1}{desc}%
2644      }%
2645      {#1}%
2646      }%
2647      {%
2648      \glsaccessdesc{#1}%
2649      }%
2650      }%
2651      }%
2652  }
2653 }
2654 {
2655 \renewcommand*{\glossentrydesc}[1]{%
2656 \glsdoifexistsorwarn{#1}%
2657 {%
2658 \glsetabbrvfmt{\glscategory{#1}}%
2659 \glsifattribute{#1}{glossdesc}{firstuc}%
2660 {%
2661 \Glsentrydesc{#1}%
2662 }%
2663 {%
2664 \glsifattribute{#1}{glossdesc}{title}%
2665 {%
2666 \@glstr@do@titlecaps@warn
2667 \glstrfieldtitlecase{#1}{desc}%
2668 }%
2669 {%
2670 \glentrydesc{#1}%
2671 }%
2672 }%
2673 }%
2674 }
2675 }

```

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

```

2676 \ifpackageloaded{glossaries-accsupp}
2677 {
2678 \renewcommand*{\glossentryname}[1]{%
2679 \glsdoifexistsorwarn{#1}%

```

```

2680 {%
2681     \glsetabbrvfmt{\glscategory{#1}}%
2682     \glusifattribute{#1}{glossname}{firstuc}%
2683     {%
2684         \glsnameaccessdisplay
2685         {%
2686             \glsnamefont{\Glsentryname{#1}}%
2687         }%
2688     {#1}%
2689 }%
2690 {%
2691     \glusifattribute{#1}{glossname}{title}%
2692     {%
2693         \@glstr@do@titlecaps@warn
2694         \glsnameaccessdisplay
2695         {%
2696             \glsnamefont{\glstrfieldtitlecase{#1}{name}}%
2697         }%
2698     {#1}%
2699 }%
2700 {%
2701     \glusifattribute{#1}{glossname}{uc}%
2702     {%
2703         \glsnameaccessdisplay
2704         {%

```

Hide the label from the upper-casing command.

```

2705         \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
2706         \glsnamefont{\mfirstucMakeUppercase{\glo@name}}%
2707     }%
2708 {#1}%
2709 }%
2710 {%
2711     \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
2712     \glsnameaccessdisplay
2713     {%
2714         \expandafter\glsnamefont\expandafter{\glo@name}%
2715     }%
2716 {#1}%
2717 }%
2718 }%
2719 }%

```

Do post-name hook:

```

2720     \glstrpostnamehook{#1}%
2721 }%
2722 }
2723 }
2724 {
2725     \renewcommand*{\glossentryname}[1]{%

```

```

2726 \glsdoifexistsorwarn{#1}%
2727 {%
2728 \glssetabbrvfmt{\glscategory{#1}}%
2729 \glsifattribute{#1}{glossname}{firstuc}%
2730 {%
2731 \glsnamefont{\Glsentryname{#1}}%
2732 }%
2733 {%
2734 \glsifattribute{#1}{glossname}{title}%
2735 {%
2736 \@glsxtr@do@titlecaps@warn
2737 \glsnamefont{\glsxtrfieldtitlecase{#1}{name}}%
2738 }%
2739 {%
2740 \glsifattribute{#1}{glossname}{uc}%
2741 {%

```

Hide the label from the upper-casing command.

```

2742 \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
2743 \glsnamefont{\mfirstucMakeUppercase{\glo@name}}%
2744 }%
2745 {%

```

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

```

2746 \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
2747 \expandafter\glsnamefont\expandafter{\glo@name}%
2748 }%
2749 }%
2750 }%
2751 }%

```

Do post-name hook:

```

2752 \glsxtrpostnamehook{#1}%
2753 }
2754 }

```

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

```

2755 \@ifpackageloaded{glossaries-accsupp}
2756 {
2757 \renewcommand*{\Glossentryname}[1]{%
2758 \glsdoifexistsorwarn{#1}%
2759 {%
2760 \glssetabbrvfmt{\glscategory{#1}}%
2761 \glsnameaccessdisplay
2762 {%
2763 \glsnamefont{\Glsentryname{#1}}%
2764 }%
2765 {#1}%

```

Do post-name hook:

```

2766     \glstrpostnamehook{#1}%
2767   }%
2768 }
2769 }
2770 {
2771   \renewcommand*{\Glossentryname}[1]{%
2772     \glsdoifexistsorwarn{#1}%
2773     {%
2774       \glssetabbrvfmt{\glscategory{#1}}%
2775       \glsnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

2776     \glstrpostnamehook{#1}%
2777   }%
2778 }
2779 }

```

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.

```

2780 \newcommand*{\glstrpostnamehook}[1]{%
2781   \def\@glsnumberformat{glsnumberformat}%
2782   \glstrdoautoindexname{#1}{indexname}%
2783 }

```

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

```

2784 \newif\if@glstr@format@override
2785 \@glstr@format@overridefalse

```

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

`xFormatOverride`

```

2786 \@ifpackageloaded{hyperref}
2787 {

```

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

```

2788   \ifHy@hyperindex
2789     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
2790       \@glstr@format@override true
2791       \appto\theindex{\let\glshypernumber\@firstofone}%
2792     }
2793   \else
2794     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
2795       \@glstr@format@override true
2796       \appto\theindex{\let\glshypernumber\hyperpage}%
2797     }

```

```

2798 \fi
2799 }
2800 {
2801 \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
2802 \@glsextr@format@overridetrue
2803 }
2804 }
2805 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

doautoindexname

```

2806 \newcommand*{\glsextrdoautoindexname}[2]{%
2807 \glshasattribute{#1}{#2}%
2808 {%

```

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.

```

2809 \@glsextr@autoindex@setname{#1}%

```

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

```

2810 \protected@edef\@glsextr@attrval{\glsggetattribute{#1}{#2}}%
2811 \if@glsextr@format@override
2812 \ifdefstring{\@glsgnumberformat}{\glsgnumberformat}{}%
2813 {\let\@glsextr@attrval\@glsgnumberformat}%
2814 \fi
2815 \ifdefstring{\@glsextr@attrval}{true}%
2816 {}%
2817 {\eappto\@glo@name{\@glsextr@autoindex@encap\@glsextr@attrval}}%
2818 \expandafter\index\expandafter{\@glo@name}%
2819 }%
2820 {}%
2821 }

```

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

```

2822 \newcommand*{\@glsextr@autoindex@setname}[1]{%
2823 \def\@glo@name{\string\glsgentryname{#1}}%
2824 \glsgletentryfield{\@glo@sort}{#1}{sort}%
2825 \@glsg@checkmkidxchars\@glo@sort
2826 \@glsextr@autoindex@doextra@esc\@glo@sort
2827 \epreto\@glo@name{\@glo@sort\@glsextr@autoindex@at}%
2828 }

```

dex@doextra@esc

```

2829 \newcommand*{\@glsextr@autoindex@doextra@esc}[1]{%

```

Escape the escape character unless it has already been escaped.

```

2830 \ifx\@glsextr@autoindex@esc\@glsgquotechar
2831 \else
2832 \def\@glsg@checkedmkidx{}%
2833 \edef\@glsextr@checkspch{%
2834 \noexpand\@glsextr@autoindex@escquote\expandonce{#1}%

```

```

2835      \noexpand\@empty\@glstr@autoindex@esc\noexpand\@nnil
2836      \@glstr@autoindex@esc\noexpand\@empty\noexpand\@glstr@endescspch}%
2837      \@glstr@checkspch
2838      \let#1\@gls@checkedmkidx\relax
2839      \fi

```

Escape actual character unless it has already been escaped.

```

2840      \ifx\@glstr@autoindex@at\@gls@actualchar
2841      \else
2842      \def\@gls@checkedmkidx{}%
2843      \edef\@glstr@checkspch{%
2844      \noexpand\@glstr@autoindex@escat\expandonce{#1}%
2845      \noexpand\@empty\@glstr@autoindex@at\noexpand\@nnil
2846      \@glstr@autoindex@at\noexpand\@empty\noexpand\@glstr@endescspch}%
2847      \@glstr@checkspch
2848      \let#1\@gls@checkedmkidx\relax
2849      \fi

```

Escape level character unless it has already been escaped.

```

2850      \ifx\@glstr@autoindex@level\@gls@levelchar
2851      \else
2852      \def\@gls@checkedmkidx{}%
2853      \edef\@glstr@checkspch{%
2854      \noexpand\@glstr@autoindex@esclevel\expandonce{#1}%
2855      \noexpand\@empty\@glstr@autoindex@level\noexpand\@nnil
2856      \@glstr@autoindex@level\noexpand\@empty\noexpand\@glstr@endescspch}%
2857      \@glstr@checkspch
2858      \let#1\@gls@checkedmkidx\relax
2859      \fi

```

Escape encap character unless it has already been escaped.

```

2860      \ifx\@glstr@autoindex@encap\@gls@encapchar
2861      \else
2862      \def\@gls@checkedmkidx{}%
2863      \edef\@glstr@checkspch{%
2864      \noexpand\@glstr@autoindex@escencap\expandonce{#1}%
2865      \noexpand\@empty\@glstr@autoindex@encap\noexpand\@nnil
2866      \@glstr@autoindex@encap\noexpand\@empty\noexpand\@glstr@endescspch}%
2867      \@glstr@checkspch
2868      \let#1\@gls@checkedmkidx\relax
2869      \fi
2870 }

```

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.

```

2871 \newcommand*{\@glstr@autoindex@at}{ }

```

trSetActualChar Set the actual character.

```

2872 \newcommand*{\GlsXtrSetActualChar}[1]{%
2873   \gdef\@glstr@autoindex@at{#1}%
2874   \def\@glstr@autoindex@escat##1##2##3\@glstr@endescspch{%
2875     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escat}{##1}{##2}{##3}%
2876   }%
2877 }
2878 \@onlypreamble\GlsXtrSetActualChar
2879 \makeatother
2880 \GlsXtrSetActualChar{@}
2881 \makeatletter

```

`\autoindex@encap` Encap character for use with `\index`.

```
2882 \newcommand*{\@glstr@autoindex@encap}{}
```

`\XtrSetEncapChar` Set the encap character.

```

2883 \newcommand*{\GlsXtrSetEncapChar}[1]{%
2884   \gdef\@glstr@autoindex@encap{#1}%
2885   \def\@glstr@autoindex@escencap##1##2##3\@glstr@endescspch{%
2886     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escencap}{##1}{##2}{##3}%
2887   }%
2888 }
2889 \GlsXtrSetEncapChar{||}
2890 \@onlypreamble\GlsXtrSetEncapChar

```

`\autoindex@level` Level character for use with `\index`.

```
2891 \newcommand*{\@glstr@autoindex@level}{}
```

`\XtrSetLevelChar` Set the encap character.

```

2892 \newcommand*{\GlsXtrSetLevelChar}[1]{%
2893   \gdef\@glstr@autoindex@level{#1}%
2894   \def\@glstr@autoindex@esclevel##1##2##3\@glstr@endescspch{%
2895     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@esclevel}{##1}{##2}{##3}%
2896   }%
2897 }
2898 \GlsXtrSetLevelChar{!}
2899 \@onlypreamble\GlsXtrSetLevelChar

```

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

```
2900 \newcommand*{\@glstr@autoindex@esc}{}
```

`\lsXtrSetEscChar` Set the escape character.

```

2901 \newcommand*{\GlsXtrSetEscChar}[1]{%
2902   \gdef\@glstr@autoindex@esc{#1}%
2903   \def\@glstr@autoindex@escquote##1##2##3\@glstr@endescspch{%
2904     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escquote}{##1}{##2}{##3}%
2905   }%
2906 }
2907 \GlsXtrSetEscChar{"}
2908 \@onlypreamble\GlsXtrSetEscChar

```


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

```
2909 \ifdef\actualchar
2910 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
2911 {}
```

Quote character \quotechar:

```
2912 \ifdef\quotechar
2913 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
2914 {}
```

Level character \levelchar:

```
2915 \ifdef\levelchar
2916 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
2917 {}
```

Encap character \encapchar:

```
2918 \ifdef\encapchar
2919 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
2920 {}
```

leto@endescspch

```
2921 \def\@glxstr@gobbleto@endescspch#1\@glxstr@endescspch{}
```

toindex@esc@spch

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

```
2922 \newcommand*{\@glxstr@autoindex@escspch}[5]{%
2923   \@glstmpb=\expandafter{\@glsc@checkedmkidx}%
2924   \toks@={#3}%
2925   \ifx\@nnil#3\relax
2926     \def\@glxstr@checkspch{\@glxstr@gobbleto@endescspch#5\@glxstr@endescspch}%
2927   \else
2928     \ifx\@nnil#4\relax
2929       \edef\@glsc@checkedmkidx{\the\@glstmpb\the\toks@}%
2930       \def\@glxstr@checkspch{\@glxstr@gobbleto@endescspch
2931         #4#5\@glxstr@endescspch}%
2932     \else
2933       \edef\@glsc@checkedmkidx{\the\@glstmpb\the\toks@
2934         \@glxstr@autoindex@esc#1}%
2935       \def\@glxstr@checkspch{#2#5#1\@nnil#1\@glxstr@endescspch}%
2936     \fi
2937   \fi
2938   \@glxstr@checkspch
2939 }
```

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

```
2940 \renewcommand*{\Glossentrydesc}[1]{%
2941   \glsoifexistsorwarn{#1}%
```

```

2942  {%
2943    \glssetabbrvfmt{\glscategory{#1}}%
2944    \Glsaccessdesc{#1}%
2945  }%
2946 }

```

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

```

2947 \renewcommand*{\glossentrysymbol}[1]{%
2948   \glsdoifexistsorwarn{#1}%
2949   {%
2950     \glssetabbrvfmt{\glscategory{#1}}%
2951     \glsaccesssymbol{#1}%
2952   }%
2953 }

```

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

```

2954 \renewcommand*{\Glossentrysymbol}[1]{%
2955   \glsdoifexistsorwarn{#1}%
2956   {%
2957     \glssetabbrvfmt{\glscategory{#1}}%
2958     \Glsaccesssymbol{#1}%
2959   }%
2960 }

```

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.

```

2961 \newcommand*{\GlsXtrEnableInitialTagging}{%
2962   \@ifstar\s@glxtr@enabletagging\@glxtr@enabletagging
2963 }
2964 \@onlypreamble\GlsXtrEnableInitialTagging

```

`\r@enabletagging` Starred version undefines command.

```

2965 \newcommand*{\s@glxtr@enabletagging}[2]{%
2966   \undef#2%
2967   \@glxtr@enabletagging{#1}{#2}%
2968 }

```

`\r@enabletagging` Internal command.

```

2969 \newcommand*{\@glxtr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
2970   \@for\@glxtr@cat:=#1\do
2971   {%
2972     \ifdefempty\@glxtr@cat
2973     {}%
2974     {\glssetcategoryattribute{\@glxtr@cat}{tagging}{true}}%

```

```

2975 }%
2976 \newrobustcmd*#2[1]{##1}%
2977 \def\@glsxtr@taggingcs{#2}%
2978 \renewcommand*\@glsxtr@activate@initialtagging{%
2979   \let#2\@glsxtr@tag
2980 }%
2981 \ifundef\@gls@preglossaryhook
2982 {\GlossariesExtraWarning{Initial tagging requires at least
2983   glossaries.sty v4.19 to work correctly}}%
2984 {}%
2985 }

```

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

```

2986 \ifundef\mfu@checkword@do
2987 {
2988   \newcommand*\mfu@checkword@do[1]{%
2989     \ifdefstring{\mfu@checkword@arg}{#1}%
2990     {%
2991       \let\@mfu@domakefirstuc\@firstofone
2992       \listbreak
2993     }%
2994   }%
2995 }

```

`\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.

```

2996 \ifundef\mfu@checkword
2997 {
2998   \newcommand{\@glsxtr@do@titlecaps@warn}{%
2999     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
3000       support not available}%

```

One warning should suffice.

```

3001   \let\@glsxtr@do@titlecaps@warn\relax
3002 }
3003 }
3004 {
3005   \renewcommand*\mfu@checkword[1]{%
3006     \def\mfu@checkword@arg{#1}%
3007     \let\@mfu@domakefirstuc\makefirstuc
3008     \forlistloop\mfu@checkword@do\@mfu@nocaplist
3009   }
3010 }
3011 }
3012 {}% no patch required

```

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

3013 \newcommand*{\@glstr@do@titlecaps@warn}{}
@initialtagging Used in \printglossary but at least v4.19 of glossaries required.
3014 \newcommand*{\@glstr@activate@initialtagging}
\@glstr@tag Definition of tagging command when used in glossary.
3015 \newrobustcmd*{\@glstr@tag}[1]{%
3016 \glsifattribute{\glscurrententrylabel}{tagging}{true}%
3017 {\glstrtagfont{#1}}{#1}%
3018 }
\glstrtagfont Used in the glossary.
3019 \newcommand*{\glstrtagfont}[1]{\underline{#1}}
preglossaryhook This macro was introduced in glossaries version 4.19, so it may not be defined. If it hasn't been defined this feature is unavailable.
3020 \ifdef\@gls@preglossaryhook
3021 {
3022 \renewcommand*{\@gls@preglossaryhook}{%
3023 \@glstr@activate@initialtagging
3024 \let\@glstr@org@postdescription\glspostdescription
3025 \renewcommand*{\glspostdescription}{%
3026 \glstrpostdescription
3027 \@glstr@org@postdescription
3028 }%
3029 }%
3030 }
3031 {}
postdescription This command will only be used if \@gls@preglossaryhook is available *and* the glossary style uses \glspostdescription without modifying it. (The nopostdesc option will suppress this.)
3032 \newcommand*{\glstrpostdescription}{%
3033 \csuse{glstrpostdesc\glscategory{\glscurrententrylabel}}%
3034 }
postdescgeneral
3035 \newcommand*{\glstrpostdescgeneral}{}
xtrpostdescterm
3036 \newcommand*{\glstrpostdescterm}{}
postdescacronym
3037 \newcommand*{\glstrpostdescacronym}{}
descabbreviation
3038 \newcommand*{\glstrpostdescabbreviation}{}

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

```
3039 \renewcommand*{\glspostlinkhook}{%
3040   \ifglstryexists{\glslabel}{\glstrpostlinkhook}{}%
3041 }
```

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

```
3042 \newcommand*{\glstrpostlinkhook}{%
3043   \glstrdiscardperiod{\glslabel}%
3044   {\glstrpostlinkendsentence}%
3045   {\glstrpostlink}%
3046 }
```

`\glstrpostlink`

```
3047 \newcommand*{\glstrpostlink}{%
3048   \csuse{glstrpostlink\glscategory{\glslabel}}%
3049 }
```

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

```
3050 \newcommand*{\glstrpostlinkendsentence}{%
3051   \ifcsdef{glstrpostlink\glscategory{\glslabel}}
3052   {%
3053     \csuse{glstrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
3054   .\spacefactor\sfcode'\. \relax
3055   }%
3056   {%
```

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

```
3057   \spacefactor\sfcode'\. \relax
3058   }%
3059 }
```

`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.

```
3060 \newcommand*{\glstrpostlinkAddDescOnFirstUse}{%
3061   \glstrifwasfirstuse{\space(\glsaccessdesc{\glslabel})}{}%
3062 }
```

`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.

```
3063 \newcommand*{\glstrpostlinkAddSymbolOnFirstUse}{%
3064   \glstrifwasfirstuse
3065   {%
```

```

3066 \ifglshassymbol{\glslabel}{\space(\glsaccesssymbol{\glslabel})}{}%
3067 }%
3068 {}%
3069 }

```

`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).

```

3070 \newcommand*\glstrdiscardperiod}[3]{%
3071 \glsifattribute{#1}{discardperiod}{true}%
3072 {%
3073 \glsifplural
3074 {%
3075 \glsifattribute{#1}{pluraldiscardperiod}{true}%
3076 {\glstrifperiod{#2}{#3}}%
3077 {#3}%
3078 }%
3079 {%
3080 \glstrifperiod{#2}{#3}%
3081 }%
3082 }%
3083 {#3}%
3084 }

```

`\glstrifperiod` Make a convenient user command to check if the next character is a full stop (period). Works like `\@ifstar` but uses `\new@ifnextchar` rather than `\@ifnextchar`

```

3085 \newcommand*\glstrifperiod}[1]{\new@ifnextchar.\@firstoftwo{#1}}

```

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

`glstr@punclist` List of characters identified as punctuation marks. (Be careful of babel shorthands!) This doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```

3086 \newcommand*\glstr@punclist{.,:;?!}

```

`punctuationmark` Add character to punctuation list.

```

3087 \newcommand*\glstraddpunctuationmark}[1]{\appto\glstr@punclist{#1}}

```

`punctuationmarks` Reset the punctuation list.

```

3088 \newcommand*\glstrsetpunctuationmarks}[1]{\def\glstr@punclist{#1}}

```

`\glstrifpunc` `\glstrifnextpunc{<true part>}{<>false part>}`

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

```

3089 \newcommand*\glstrifnextpunc}[2]{%

```

```

3090 \def\reserved@a{#1}%
3091 \def\reserved@b{#2}%
3092 \futurelet\@glspunc@token\glxtr@ifnextpunc
3093 }

```

`glxtr@ifnextpunc`

```

3094 \newcommand*\glxtr@ifnextpunc{%
3095 \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
3096 \reserved@b
3097 }

```

`glxtr@ifpunctoken` Test if the token given in the first argument is in the punctuation list.

```

3098 \newcommand*\glxtr@ifpunctoken[1]{%
3099 \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
3100 }

```

`glxtr@ifpunctoken`

```

3101 \def\@glxtr@ifpunctoken#1#2{%
3102 \let\reserved@d=#2%
3103 \ifx\reserved@d\@nnil
3104 \let\glxtr@next\@glxtr@notfoundinlist
3105 \else
3106 \ifx#1\reserved@d
3107 \let\glxtr@next\@glxtr@foundinlist
3108 \else
3109 \let\glxtr@next\@glxtr@ifpunctoken
3110 \fi
3111 \fi
3112 \glxtr@next#1%
3113 }

```

`glxtr@foundinlist`

```

3114 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}

```

`glxtr@notfoundinlist`

```

3115 \def\@glxtr@notfoundinlist#1{\@secondoftwo}

```

`glxtr@dopostpunc`

```
\glxtr@dopostpunc{<code>}
```

If this is followed by a punctuation character, do *<code>* after the character otherwise do *<code>* before whatever comes next.

```

3116 \newcommand*\glxtr@dopostpunc[1]{%
3117 \glxtr@ifnextpunc{\@glxtr@swaptwo{#1}}{#1}%
3118 }

```

`glxtr@swaptwo`

```

3119 \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.

```
3120 \define@key{glstrabbrv}{category}{%
3121   \edef\glscategorylabel{#1}%
3122   \ifcsdef{@glssabbrv@current@#1}%
3123     {%
3124       \glstr@applyabbrvstyle{\csname @glssabbrv@current@#1\endcsname}%
3125     }%
3126   }%
3127 }
```

Save the short plural form. This may be needed before the entry is defined.

```
3128 \define@key{glstrabbrv}{shortplural}{%
3129   \def\@gls@shortpl{#1}%
3130 }
```

Similarly for the long plural form.

```
3131 \define@key{glstrabbrv}{longplural}{%
3132   \def\@gls@longpl{#1}%
3133 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```
3134 \newtoks\glsshortpltok
```

\glslongpltok

```
3135 \newtoks\glslongpltok
```

glstr@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.

```
3136 \newcommand*{\@glstr@insertdots}[2]{%
3137   \def#1{%
3138     \@glstr@insert@dots#1#2\@nnil
3139   }
```

glstr@insert@dots

```
3140 \newcommand*{\@glstr@insert@dots}[2]{%
3141   \ifx\@nnil#2\relax
```



```

3142 \let\@glsxtr@insert@dots@next\@gobble
3143 \else
3144 \ifx\relax#2\relax
3145 \else
3146 \appto#1{#2.}%
3147 \fi
3148 \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
3149 \fi
3150 \@glsxtr@insert@dots@next#1%
3151 }

```

newabbreviation Define a new generic abbreviation.

```

3152 \newcommand*{\newabbreviation}[4] [] {%
3153 \glskeylisttok{#1}%
3154 \glslabeltok{#2}%
3155 \glsshorttok{#3}%
3156 \glslongtok{#4}%

  Get the category.
3157 \def\glscategorylabel{abbreviation}%
3158 \glsxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
3159 \setkeys*{glsxtrabbrv}[shortplural,longplural]{#1}%

  Set the default long plural
3160 \def\@gls@longpl{#4\glspluralsuffix}%

  Has the insertdots attribute been set?
3161 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
3162 {%
3163 \@glsxtr@insertdots\@gls@short{#3}%
3164 \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
3165 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
3166 {%
3167 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
3168 '\abbrvpluralsuffix}%
3169 }%
3170 {%
3171 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
3172 {%
3173 \let\@gls@shortpl\@gls@short
3174 }%
3175 {%
3176 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
3177 \abbrvpluralsuffix}%
3178 }%
3179 }%
3180 }%
3181 {%

  insertdots not true.
3182 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%

```

```

3183   {%
3184     \def\@gls@shortpl{#3'\abbrvpluralsuffix}%
3185   }%
3186   {%
3187     \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
3188     {%
3189       \def\@gls@shortpl{#3}%
3190     }%
3191     {%
3192       \def\@gls@shortpl{#3\abbrvpluralsuffix}%
3193     }%
3194   }%
3195 }%

```

Hook for further customisation if required:

```

3196 \glsxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%

```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary.

```

3197 \setkeys*{\glsxtrabbrv}[category]{#1}%

```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```

3198 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
3199 \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

3200 \newabbreviationhook

```

Define this entry:

```

3201 \protected@edef\@do@newglossaryentry{%
3202   \noexpand\newglossaryentry{\the\glslabeltok}%
3203   {%
3204     type=\glsxtrabbrvtype,%
3205     category=abbreviation,%
3206     short={\the\glsshorttok},%
3207     shortplural={\the\glsshortpltok},%
3208     long={\the\glslongtok},%
3209     longplural={\the\glslongpltok},%
3210     name={\the\glsshorttok},%
3211     \CustomAbbreviationFields,%
3212     \the\glskeylisttok
3213   }%
3214 }%
3215 \@do@newglossaryentry
3216 \GlsXtrPostNewAbbreviation
3217 }

```

`\evpresetkeyhook` Hook for extra stuff in `\newabbreviation`

```

3218 \newcommand*{\glsxtrnewabbrevpresetkeyhook}[3]{}

```

`\NewAbbreviation` Hook used by abbreviation styles.

```

3219 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

`\newabbreviationhook` Hook for use with `\newabbreviation`.

```
3220 \newcommand*{\newabbreviationhook}{}

```

`\CustomAbbreviationFields`

```
3221 \newcommand*{\CustomAbbreviationFields}{}

```

`\glstrfullformat` Full format without case change.

```
3222 \newcommand*{\glstrfullformat}[2]{%
3223   \glstrfirstlongfont{\glstraccesslong{#1}}#2\glstrfullsep{#1}%
3224   (\protect\glstrfirstabbrvfont{\glstraccessshort{#1}})%
3225 }

```

`\Glsstrfullformat` Full format with case change.

```
3226 \newcommand*{\Glsstrfullformat}[2]{%
3227   \glstrfirstlongfont{\Glsstraccesslong{#1}}#2\glstrfullsep{#1}%
3228   (\protect\glstrfirstabbrvfont{\Glsstraccessshort{#1}})%
3229 }

```

`\glstrfullplformat` Plural full format without case change.

```
3230 \newcommand*{\glstrfullplformat}[2]{%
3231   \glstrfirstlongfont{\glstraccesslongpl{#1}}#2\glstrfullsep{#1}%
3232   (\protect\glstrfirstabbrvfont{\glstraccessshortpl{#1}})%
3233 }

```

`\Glsstrfullplformat` Plural full format with case change.

```
3234 \newcommand*{\Glsstrfullplformat}[2]{%
3235   \glstrfirstlongfont{\Glsstraccesslongpl{#1}}#2\glstrfullsep{#1}%
3236   (\protect\glstrfirstabbrvfont{\Glsstraccessshortpl{#1}})%
3237 }

```

`\glstrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
3238 \newcommand*{\glstrfullsep}[1]{\space}

```

In-line formats in case first use isn't compatible with `\glstrentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glstrinlinefullformat` Full format without case change.

```
3239 \newcommand*{\glstrinlinefullformat}{\glstrfullformat}

```

`\Glsstrinlinefullformat` Full format with case change.

```
3240 \newcommand*{\Glsstrinlinefullformat}{\Glsstrfullformat}

```

`\glstrfullplformat` Plural full format without case change.

```
3241 \newcommand*{\glstrinlinefullplformat}{\glstrfullplformat}

```

`\Glsstrinlinefullplformat` Plural full format with case change.

```
3242 \newcommand*{\Glsstrinlinefullplformat}{\Glsstrfullplformat}

```

Redefine `\glsentryfull` etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the `\glsxtrfull` set of commands instead.

```

\glsentryfull
3243 \renewcommand*{\glsentryfull}[1]{\glsxtrinlinefullformat{#1}{}}

\Glsentryfull
3244 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}

\glsentryfullpl
3245 \renewcommand*{\glsentryfullpl}[1]{\glsxtrinlinefullplformat{#1}{}}

\Glsentryfullpl
3246 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}

\firstabbrvfont  Font changing command used for the abbreviation on first use or in the full format.
3247 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}

\abbrvdefaultfont  Font changing command used for the abbreviation on first use or in the full format.
3248 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}

\glsabbrvfont  Font changing command used for the abbreviation on subsequent use.
3249 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}

\abbrvdefaultfont
3250 \newcommand*{\glsabbrvdefaultfont}[1]{#1}

\firstlongfont  Font changing command used for the long form on first use or in the full format.
3251 \newcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{#1}}

\longdefaultfont
3252 \newcommand*{\glsfirstlongdefaultfont}[1]{#1}

\brvpluralsuffix  Default plural suffix.
3253 \newcommand*{\abbrvpluralsuffix}{\glspluralsuffix}

\glsxtrfull  Full form (no case-change).
3254 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\ns@glsxtrfull}
3255 \newcommand*{\ns@glsxtrfull}[2][{}]{%
3256   \new@ifnextchar[{\@glsxtr@full{#1}{#2}}{%
3257     {\@glsxtr@full{#1}{#2}[]}%
3258 }

```

`\@glxtr@full` Low-level macro:

```
3259 \def\@glxtr@full#1#2[#3]{%
3260   \glstoifexists{#2}%
3261   {%
3262     \glsetabbrvfmt{\glscategory{#2}}%
3263     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3264     \let\gl@ifplural\@secondoftwo
3265     \let\glscapscase\@firstofthree
3266     \let\glinsert\@empty
3267     \def\glscustomtext{\glxtrinlinefullformat{#2}{#3}}%
```

What should `\glxtrifwasfirstuse` be set to here? Where the inline and display full forms are the same, this is essentially emulating first use, to it make sense for the postlink hook to pretend it was a first use instance. It makes less sense if the inline and display forms are different. Provide a hook to make it easier to reconfigure.

```
3268   \glxtrsetupfulldefs
3269   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3270   }%
3271   \glspostlinkhook
3272 }
```

`trsetupfulldefs`

```
3273 \newcommand*{\glxtrsetupfulldefs}{%
3274   \let\glxtrifwasfirstuse\@firstoftwo
3275 }
```

`\Glsxtrfull` Full form (first letter uppercase).

```
3276 \newrobustcmd*{\Glsxtrfull}{\@gl@hyp@opt\ns@Glsxtrfull}
3277 \newcommand*\ns@Glsxtrfull[2][ ]{%
3278   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}%
3279   {\@Glsxtr@full{#1}{#2} [ ]}%
3280 }
```

`\@Glsxtr@full` Low-level macro:

```
3281 \def\@Glsxtr@full#1#2[#3]{%
3282   \glstoifexists{#2}%
3283   {%
3284     \glsetabbrvfmt{\glscategory{#2}}%
3285     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3286     \let\gl@ifplural\@secondoftwo
3287     \let\glscapscase\@secondofthree
3288     \let\glinsert\@empty
3289     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
3290     \glxtrsetupfulldefs
3291     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3292     }%
3293     \glspostlinkhook
3294 }
```

`\GLSxtrfull` Full form (all uppercase).

```
3295 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\ns@GLSxtrfull}
3296 \newcommand*\ns@GLSxtrfull[2][\]{%
3297   \new@ifnextchar[\@GLSxtr@full{#1}{#2}}%
3298   {\@GLSxtr@full{#1}{#2}[]}%
3299 }
```

`\@GLSxtr@full` Low-level macro:

```
3300 \def\@GLSxtr@full#1#2[#3]{%
3301   \glsdoifexists{#2}%
3302   {%
3303     \glssetabbrvfmt{\glscategory{#2}}%
3304     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3305     \let\glsifplural\@secondoftwo
3306     \let\glsapscase\@thirdofthree
3307     \let\glsinsert\@empty
3308     \def\glscustomtext{\mfirstucMakeUppercase{\glsxtrinlinefullformat{#2}{#3}}}%
3309     \glsxtrsetupfulldefs
3310     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3311   }%
3312   \glspostlinkhook
3313 }
```

`\glsxtrfullpl` Plural full form (no case-change).

```
3314 \newrobustcmd*{\glsxtrfullpl}{\@gls@hyp@opt\ns@glsxtrfullpl}
3315 \newcommand*\ns@glsxtrfullpl[2][\]{%
3316   \new@ifnextchar[\@glsxtr@fullpl{#1}{#2}}%
3317   {\@glsxtr@fullpl{#1}{#2}[]}%
3318 }
```

`\@glsxtr@fullpl` Low-level macro:

```
3319 \def\@glsxtr@fullpl#1#2[#3]{%
3320   \glsdoifexists{#2}%
3321   {%
3322     \glssetabbrvfmt{\glscategory{#2}}%
3323     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3324     \let\glsifplural\@firstoftwo
3325     \let\glsapscase\@firstofthree
3326     \let\glsinsert\@empty
3327     \def\glscustomtext{\glsxtrinlinefullplformat{#2}{#3}}%
3328     \glsxtrsetupfulldefs
3329     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3330   }%
3331   \glspostlinkhook
3332 }
```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```
3333 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\ns@Glsxtrfullpl}
3334 \newcommand*\ns@Glsxtrfullpl[2][\]{%
```

```

3335 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
3336           {\@Glsxtr@fullpl{#1}{#2}[]}%
3337 }

```

\@Glsxtr@fullpl Low-level macro:

```

3338 \def\@Glsxtr@fullpl#1#2[#3]{%
3339   \glsdoifexists{#2}%
3340   {%
3341     \glssetabbrvfmt{\glscategory{#2}}%
3342     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3343     \let\glsifplural\@firstoftwo
3344     \let\glscapscase\@secondofthree
3345     \let\glsinsert\@empty
3346     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
3347     \glsxtrsetupfulldefs
3348     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3349   }%
3350   \glspostlinkhook
3351 }

```

\Glsxtrfullpl Plural full form (all upper case).

```

3352 \newrobustcmd*{\Glsxtrfullpl}{\@gl@hyp@opt\@ns@Glsxtrfullpl}
3353 \newcommand*\ns@Glsxtrfullpl[2][]{%
3354   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
3355           {\@Glsxtr@fullpl{#1}{#2}[]}%
3356 }

```

\@Glsxtr@fullpl Low-level macro:

```

3357 \def\@Glsxtr@fullpl#1#2[#3]{%
3358   \glsdoifexists{#2}%
3359   {%
3360     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3361     \let\glsifplural\@firstoftwo
3362     \let\glscapscase\@thirdofthree
3363     \let\glsinsert\@empty
3364     \def\glscustomtext{%
3365       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
3366     \glsxtrsetupfulldefs
3367     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3368   }%
3369   \glspostlinkhook
3370 }

```

The short and long forms work in a similar way to acronyms.

\glsxtrshort

```

3371 \newrobustcmd*{\glsxtrshort}{\@gl@hyp@opt\@ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3372 \newcommand*{\ns@glxstrshort}[2] [] {%
3373   \new@ifnextchar[{\@glxstrshort{#1}{#2}}{\@glxstrshort{#1}{#2} [] }%
3374 }

```

Read in the final optional argument:

```

3375 \def\@glxstrshort#1#2[#3] {%
3376   \glsdoifexists{#2}%
3377   {%

```

Need to make sure \glsabbrvfont is set correctly.

```

3378     \glssetabbrvfmt{\glscategory{#2}}%
3379     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3380     \let\glxtrifwasfirstuse\@secondoftwo
3381     \let\glsifplural\@secondoftwo
3382     \let\glscapscase\@firstofthree
3383     \let\glsinsert\@empty
3384     \def\glscustomtext{%
3385       \glsabbrvfont{\glsaccessshort{#2}}#3%
3386     }%
3387     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3388   }%
3389   \glspostlinkhook
3390 }

```

\Glsxtrshort

```

3391 \newrobustcmd*{\Glsxtrshort}{\@gl@hyp@opt\ns@Glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3392 \newcommand*{\ns@Glsxtrshort}[2] [] {%
3393   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} [] }%
3394 }

```

Read in the final optional argument:

```

3395 \def\@Glsxtrshort#1#2[#3] {%
3396   \glsdoifexists{#2}%
3397   {%
3398     \glssetabbrvfmt{\glscategory{#2}}%
3399     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3400     \let\glxtrifwasfirstuse\@secondoftwo
3401     \let\glsifplural\@secondoftwo
3402     \let\glscapscase\@secondofthree
3403     \let\glsinsert\@empty
3404     \def\glscustomtext{%
3405       \glsabbrvfont{\Glsaccessshort{#2}}#3%
3406     }%
3407     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3408   }%
3409   \glspostlinkhook
3410 }

```

\GLSxtrshort


```

3411 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\ns@GLSxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
3412 \newcommand*{\ns@GLSxtrshort}[2] [] {%
3413   \new@ifnextchar[\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} []}%
3414 }

    Read in the final optional argument:
3415 \def\@GLSxtrshort#1#2[#3] {%
3416   \glsdoifexists{#2}%
3417   {%
3418     \glssetabbrvfmt{\glscategory{#2}}%
3419     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3420     \let\glsxtrifwasfirstuse\@secondoftwo
3421     \let\glsifplural\@secondoftwo
3422     \let\glsapscase\@thirdofthree
3423     \let\glsinsert\@empty
3424     \def\glscustomtext{%
3425       \mfirstucMakeUppercase{\glsabbrvfont{\glsaccessshort{#2}}#3}%
3426     }%
3427     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3428   }%
3429   \glspostlinkhook
3430 }

```

\glsxtrlong

```

3431 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
3432 \newcommand*{\ns@glsxtrlong}[2] [] {%
3433   \new@ifnextchar[\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2} []}%
3434 }

    Read in the final optional argument:
3435 \def\@glsxtrlong#1#2[#3] {%
3436   \glsdoifexists{#2}%
3437   {%
3438     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3439     \let\glsxtrifwasfirstuse\@secondoftwo
3440     \let\glsifplural\@secondoftwo
3441     \let\glsapscase\@firstofthree
3442     \let\glsinsert\@empty
3443     \def\glscustomtext{\glsaccesslong{#2}#3}%
3444     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3445   }%
3446   \glspostlinkhook
3447 }

```

\Glsxtrlong

```

3448 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\ns@Glsxtrlong}

```

Define the un-starred form. Need to determine if there is a final optional argument

```
3449 \newcommand*{\ns@Glsxtrlong}[2] [] {%
3450   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
3451 }
```

Read in the final optional argument:

```
3452 \def\@Glsxtrlong#1#2[#3] {%
3453   \glsdoifexists{#2}%
3454   {%
3455     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3456     \let\glxtrifwasfirstuse\@secondoftwo
3457     \let\gl@sifplural\@secondoftwo
3458     \let\glscapscase\@secondofthree
3459     \let\gl$insert\@empty
3460     \def\glscustomtext{\Glsaccesslong{#2}#3}%
3461     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3462   }%
3463   \glspostlinkhook
3464 }
```

\GLSxtrlong

```
3465 \newrobustcmd*{\GLSxtrlong}{\@gl@hyp@opt\ns@GLSxtrlong}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3466 \newcommand*{\ns@GLSxtrlong}[2] [] {%
3467   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
3468 }
```

Read in the final optional argument:

```
3469 \def\@GLSxtrlong#1#2[#3] {%
3470   \glsdoifexists{#2}%
3471   {%
3472     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3473     \let\glxtrifwasfirstuse\@secondoftwo
3474     \let\gl@sifplural\@secondoftwo
3475     \let\glscapscase\@thirdofthree
3476     \let\gl$insert\@empty
3477     \def\glscustomtext{\mfirstucMakeUppercase{\Glsaccesslong{#2}#3}}%
3478     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3479   }%
3480   \glspostlinkhook
3481 }
```

Plural short forms:

\glxtrshortpl

```
3482 \newrobustcmd*{\glxtrshortpl}{\@gl@hyp@opt\ns@glxtrshortpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3483 \newcommand*{\ns@glxtrshortpl}[2] [] {%
3484   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2} []}%
3485 }
```

Read in the final optional argument:

```

3486 \def\@glstrshortpl#1#2[#3]{%
3487   \glstoifexists{#2}%
3488   {%
3489     \glsssetabbrvfmt{\glscategory{#2}}%
3490     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3491     \let\glstrifwasfirstuse\@secondoftwo
3492     \let\gl@ifplural\@firstoftwo
3493     \let\glscapscase\@firstofthree
3494     \let\glinsert\@empty
3495     \def\glscustomtext{%
3496       \glssabbrvfont{\Glsaccessshortpl{#2}}#3%
3497     }%
3498     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3499   }%
3500   \glspostlinkhook
3501 }

```

\Glsxtrshortpl

```

3502 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\ns@Glsxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3503 \newcommand*{\ns@Glsxtrshortpl}[2][ ]{%
3504   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} [ ]}%
3505 }

```

Read in the final optional argument:

```

3506 \def\@Glsxtrshortpl#1#2[#3]{%
3507   \glstoifexists{#2}%
3508   {%
3509     \glsssetabbrvfmt{\glscategory{#2}}%
3510     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3511     \let\glstrifwasfirstuse\@secondoftwo
3512     \let\gl@ifplural\@firstoftwo
3513     \let\glscapscase\@secondofthree
3514     \let\glinsert\@empty
3515     \def\glscustomtext{%
3516       \Glsabbrvfont{\Glsaccessshortpl{#2}}#3%
3517     }%
3518     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3519   }%
3520   \glspostlinkhook
3521 }

```

\GLSxtrshortpl

```

3522 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\ns@GLSxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3523 \newcommand*{\ns@GLSxtrshortpl}[2][ ]{%
3524   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} [ ]}%
3525 }

```

Read in the final optional argument:

```

3526 \def\@GLSxtrshortpl#1#2[#3]{%
3527   \glsdoifexists{#2}%
3528   {%
3529     \glsetabbrvfmt{\glscategory{#2}}%
3530     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3531     \let\glxtrifwasfirstuse\@secondoftwo
3532     \let\glsifplural\@firstoftwo
3533     \let\glscapscase\@thirdofthree
3534     \let\glsinsert\@empty
3535     \def\glscustomtext{%
3536       \mfirstucMakeUppercase{\glsabbrvfont{\glsaccessshortpl{#2}}#3}%
3537     }%
3538     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3539   }%
3540   \glspostlinkhook
3541 }

```

Plural long forms:

\glsxtrlongpl

```

3542 \newrobustcmd*{\glsxtrlongpl}{\@gl@hyp@opt\@ns@glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3543 \newcommand*{\ns@glsxtrlongpl}[2][ ]{%
3544   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2}[]}%
3545 }

```

Read in the final optional argument:

```

3546 \def\@glsxtrlongpl#1#2[#3]{%
3547   \glsdoifexists{#2}%
3548   {%
3549     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3550     \let\glxtrifwasfirstuse\@secondoftwo
3551     \let\glsifplural\@firstoftwo
3552     \let\glscapscase\@firstofthree
3553     \let\glsinsert\@empty
3554     \def\glscustomtext{\glsaccesslongpl{#2}#3}%
3555     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3556   }%
3557   \glspostlinkhook
3558 }

```

\Glsxtrlongpl

```

3559 \newrobustcmd*{\Glsxtrlongpl}{\@gl@hyp@opt\@ns@Glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3560 \newcommand*{\ns@Glsxtrlongpl}[2][ ]{%
3561   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2}[]}%
3562 }

```

Read in the final optional argument:

```

3563 \def\@GLSxtrlongpl#1#2[#3]{%
3564   \glsdoifexists{#2}%
3565   {%
3566     \let\do@glS@link@checkfirsthyper\@glS@link@nocheckfirsthyper
3567     \let\glSxtrifwasfirstuse\@secondoftwo
3568     \let\glSifplural\@firstoftwo
3569     \let\glScapscase\@secondofthree
3570     \let\glSinsert\@empty
3571     \def\glScustomtext{\@glSaccesslongpl{#2}#3}%
3572     \@glS@link[#1]{#2}{\csname glS@\glStype @entryfmt\endcsname}%
3573   }%
3574   \glSpostlinkhook
3575 }

```

\GLSxtrlongpl

```

3576 \newrobustcmd*{\GLSxtrlongpl}{\@glS@hyp@opt\@ns@GLSxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3577 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
3578   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
3579 }

```

Read in the final optional argument:

```

3580 \def\@GLSxtrlongpl#1#2[#3]{%
3581   \glsdoifexists{#2}%
3582   {%
3583     \let\do@glS@link@checkfirsthyper\@glS@link@nocheckfirsthyper
3584     \let\glSxtrifwasfirstuse\@secondoftwo
3585     \let\glSifplural\@firstoftwo
3586     \let\glScapscase\@thirdofthree
3587     \let\glSinsert\@empty
3588     \def\glScustomtext{\mfirstucMakeUppercase{\@glSaccesslongpl{#2}#3}}%
3589     \@glS@link[#1]{#2}{\csname glS@\glStype @entryfmt\endcsname}%
3590   }%
3591   \glSpostlinkhook
3592 }

```

\glSsetabbrvfmt Set the current format for the given category (or the abbreviation category if unset).

```

3593 \newcommand*{\glSsetabbrvfmt}[1]{%
3594   \ifcsdef{@glSabbrv@current@#1}%
3595     {\glSxtr@applyabbrvfmt{\csname @glSabbrv@current@#1\endcsname}}%
3596     {\glSxtr@applyabbrvfmt{@glSabbrv@current@abbreviation}}%
3597 }

```

\glSxtrgenabbrvfmt Similar to \glSgenacfmt, but for abbreviations.

```

3598 \newcommand*{\glSxtrgenabbrvfmt}{%
3599   \ifdefempty\glScustomtext
3600   {%

```

3601 \ifglsused\glslabel
 3602 {%

Subsequent use:

3603 \glsifplural
 3604 {%

Subsequent plural form:

3605 \glscapscase
 3606 {%

Subsequent plural form, don't adjust case:

3607 \glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert
 3608 }%
 3609 {%

Subsequent plural form, make first letter upper case:

3610 \glsabbrvfont{\Glsaccessshortpl{\glslabel}}\glsinsert
 3611 }%
 3612 {%

Subsequent plural form, all caps:

3613 \mfirstucMakeUppercase
 3614 {\glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert}%
 3615 }%
 3616 }%
 3617 {%

Subsequent singular form

3618 \glscapscase
 3619 {%

Subsequent singular form, don't adjust case:

3620 \glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert
 3621 }%
 3622 {%

Subsequent singular form, make first letter upper case:

3623 \glsabbrvfont{\Glsaccessshort{\glslabel}}\glsinsert
 3624 }%
 3625 {%

Subsequent singular form, all caps:

3626 \mfirstucMakeUppercase
 3627 {\glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert}%
 3628 }%
 3629 }%
 3630 }%
 3631 {%

First use:

3632 \glsifplural
 3633 {%

First use plural form:

```
3634      \glscapscase
3635      {%
```

First use plural form, don't adjust case:

```
3636      \glxtrfullplformat{\glslabel}{\glsinsert}%
3637      }%
3638      {%
```

First use plural form, make first letter upper case:

```
3639      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
3640      }%
3641      {%
```

First use plural form, all caps:

```
3642      \mfirstucMakeUppercase
3643      {\glxtrfullplformat{\glslabel}{\glsinsert}}%
3644      }%
3645      }%
3646      {%
```

First use singular form

```
3647      \glscapscase
3648      {%
```

First use singular form, don't adjust case:

```
3649      \glxtrfullformat{\glslabel}{\glsinsert}%
3650      }%
3651      {%
```

First use singular form, make first letter upper case:

```
3652      \Glsxtrfullformat{\glslabel}{\glsinsert}%
3653      }%
3654      {%
```

First use singular form, all caps:

```
3655      \mfirstucMakeUppercase
3656      {\glxtrfullformat{\glslabel}{\glsinsert}}%
3657      }%
3658      }%
3659      }%
3660      }%
3661      {%
```

User supplied text.

```
3662      \glscustomtext
3663      }%
3664 }
```

1.6.1 Abbreviation Styles Setup

`abbreviationstyle`

```

3665 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
3666   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
3667   {%
3668     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
3669   }%
3670   {%

```

Have abbreviations already been defined for this category?

```

3671   \ifcsstring{@glsabbrv@current@#1}{#2}%
3672   {%

```

Style already set.

```

3673   }%
3674   {%
3675     \def\@glsxtr@dostylewarn{%
3676       \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
3677       {%
3678         \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
3679           style has been switched \MessageBreak
3680           for category ‘#1’, \MessageBreak
3681           but there have already been entries \MessageBreak
3682           defined for this category. Unwanted \MessageBreak
3683           side-effects may result}}%
3684         \@endfortrue
3685       }%
3686       \@glsxtr@dostylewarn

```

Set up the style for the given category.

```

3687       \csdef{@glsabbrv@current@#1}{#2}%
3688       \glsxtr@applyabbrvstyle{#2}%
3689     }%
3690   }%
3691 }

```

`applyabbrvstyle` Apply the abbreviation style without existence check.

```

3692 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
3693   \csuse{@glsabbrv@dispstyle@setup@#1}%
3694   \csuse{@glsabbrv@dispstyle@fmts@#1}%
3695 }

```

`r@applyabbrvfmt` Only apply the style formats.

```

3696 \newcommand*{\glsxtr@applyabbrvfmt}[1]{%
3697   \csuse{@glsabbrv@dispstyle@fmts@#1}%
3698 }

```

`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.


```

3699 \newcommand*{\newabbreviationstyle}[3]{%
3700   \ifcsdef{@glsabbrv@dispstyle@#1}
3701   {%
3702     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
3703       defined}{}}%
3704   }%
3705   {%
3706     \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```

3707     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
3708     #2}%
3709     \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```

3710     \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
3711     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
3712     \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
3713     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
3714     #3}%
3715   }%
3716 }

```

eAbbrStyleSetup

```

3717 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
3718   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
3719   {%
3720     \PackageError{glossaries-extra}%
3721     {Unknown abbreviation style definitions ‘#1’}{}}%
3722   }%
3723   {%
3724     \csname @glsabbrv@dispstyle@setup@#1\endcsname
3725   }%
3726 }

```

seAbbrStyleFmt

```

3727 \newcommand*{\GlsXtrUseAbbrStyleFmt}[1]{%
3728   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
3729   {%
3730     \PackageError{glossaries-extra}%
3731     {Unknown abbreviation style formats ‘#1’}{}}%
3732   }%
3733   {%
3734     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
3735   }%
3736 }

```

1.6.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn't set to "true". If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glstrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

long-short

```
3737 \newabbreviationstyle{long-short}%
3738 {%
3739   \renewcommand*{\CustomAbbreviationFields}{%
3740     name={\protect\glsabbrvfont{\the\glsshorttok}},
3741     sort={\the\glsshorttok},
3742     first={\protect\glsfirstlongfont{\the\glslongtok}%
3743       \protect\glstrfullsep{\the\glslabeltok}%
3744       (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
3745     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
3746       \protect\glstrfullsep{\the\glslabeltok}%
3747       (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%
3748     plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
3749     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
3750 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3751   \glshasattribute{\the\glslabeltok}{regular}%
3752   {%
3753     \glissetattribute{\the\glslabeltok}{regular}{false}%
3754     }%
3755   }%
3756 }%
3757 }%
3758 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
3759 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
3760 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
3761 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
3762 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
3763 \renewcommand*{\glstrfullformat}[2]{%
3764   \glsfirstlongfont{\glsaccesslong{##1}}##2\glstrfullsep{##1}%
3765   (\glsfirstabbrvfont{\glsaccessshort{##1}})%
3766 }%
3767 \renewcommand*{\glstrfullplformat}[2]{%
3768   \glsfirstlongfont{\glsaccesslongpl{##1}}##2\glstrfullsep{##1}%
3769   (\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
```

```

3770 }%
3771 \renewcommand*{\Glsxtrfullformat}[2]{%
3772   \glsfirstlongfont{\Glsaccesslong{##1}}##2\glsxtrfullsep{##1}%
3773   (\glsfirstabbrvfont{\Glsaccessshort{##1}})%
3774 }%
3775 \renewcommand*{\Glsxtrfullplformat}[2]{%
3776   \glsfirstlongfont{\Glsaccesslongpl{##1}}##2\glsxtrfullsep{##1}%
3777   (\glsfirstabbrvfont{\Glsaccessshortpl{##1}})%
3778 }%
3779 }

```

Set this as the default style for general abbreviations:

```

3780 \setabbreviationstyle{long-short}

```

long-short-desc User supplies description. The long form is included in the name.

```

3781 \newabbreviationstyle{long-short-desc}%
3782 {%
3783   \renewcommand*{\CustomAbbreviationFields}{%
3784     name={\protect\glsxtrfullformat{\the\glslabeltok}{}},
3785     sort={\the\glsshorttok},%
3786     first={\protect\glsfirstlongfont{\the\glslongtok}%
3787       \protect\glsxtrfullsep{\the\glslabeltok}%
3788       (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
3789     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
3790       \protect\glsxtrfullsep{\the\glslabeltok}%
3791       (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%
3792     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

3793   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3794     \glshasattribute{\the\glslabeltok}{regular}%
3795     {%
3796       \glssetattribute{\the\glslabeltok}{regular}{false}%
3797     }%
3798   }%
3799 }%
3800 }%
3801 {%
3802   \GlsXtrUseAbbrStyleFmts{long-short}%
3803 }

```

short-long Short form followed by long form in parenthesis on first use.

```

3804 \newabbreviationstyle{short-long}%
3805 {%
3806   \renewcommand*{\CustomAbbreviationFields}{%
3807     name={\protect\glsabbrvfont{\the\glsshorttok}},
3808     sort={\the\glsshorttok},
3809     description={\the\glslongtok},%
3810     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
3811       \protect\glsxtrfullsep{\the\glslabeltok}%

```

```

3812     (\protect\glsfirstlongfont{\the\glslongtok}))},%
3813     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
3814     \protect\glsxtrfullsep{\the\glslabeltok}%
3815     (\protect\glsfirstlongfont{\the\glslongpltok}))},%
3816     plural={\protect\glsabbvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

3817 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3818     \glshasattribute{\the\glslabeltok}{regular}%
3819     {%
3820         \glssetattribute{\the\glslabeltok}{regular}{false}%
3821     }%
3822     {}%
3823 }%
3824 }%
3825 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

3826 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
3827 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvdefaultfont{##1}}%
3828 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
3829 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

3830 \renewcommand*{\glsxtrfullformat}[2]{%
3831     \glsfirstabbrvfont{\glsaccessshort{##1}}##2\glsxtrfullsep{##1}%
3832     (\glsfirstlongfont{\glsaccesslong{##1}})%
3833 }%
3834 \renewcommand*{\glsxtrfullplformat}[2]{%
3835     \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2\glsxtrfullsep{##1}%
3836     (\glsfirstlongfont{\glsaccesslongpl{##1}})%
3837 }%
3838 \renewcommand*{\Glsxtrfullformat}[2]{%
3839     \glsfirstabbrvfont{\Glsaccessshort{##1}}##2\glsxtrfullsep{##1}%
3840     (\glsfirstlongfont{\Glsaccesslong{##1}})%
3841 }%
3842 \renewcommand*{\Glsxtrfullplformat}[2]{%
3843     \glsfirstabbrvfont{\Glsaccessshortpl{##1}}##2\glsxtrfullsep{##1}%
3844     (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
3845 }%
3846 }

```

short-long-desc User supplies description. The long form is included in the name.

```

3847 \newabbreviationstyle{short-long-desc}%
3848 {%
3849     \renewcommand*{\CustomAbbreviationFields}{%
3850         name={\protect\glsxtrfullformat{\the\glslabeltok}{}},
3851         sort={\the\glsshorttok},%
3852         first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
3853         \protect\glsxtrfullsep{\the\glslabeltok}%

```

```

3854     (\protect\glsfirstlongfont{\the\glslongtok}}),%
3855     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
3856     \protect\glsxtrfullsep{\the\glslabeltok}%
3857     (\protect\glsfirstlongfont{\the\glslongpltok}}),%
3858     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

3859 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3860   \glshasattribute{\the\glslabeltok}{regular}%
3861   {%
3862     \glissetattribute{\the\glslabeltok}{regular}{false}%
3863   }%
3864   {}%
3865 }%
3866}%
3867{%
3868 \GlsXtrUseAbbrStyleFmts{short-long}%
3869 }

```

footnote Short form followed by long form in footnote on first use. Take care about using `\glsfirst` as this won't suppress the hyperlink. (Perhaps modify `\glsfirst` to reflect `nohyperfirst` attribute?)

```

3870 \newabbreviationstyle{footnote}%
3871 {%
3872   \renewcommand*{\CustomAbbreviationFields}{%
3873     name={\protect\glsabbrvfont{\the\glsshortttok}},
3874     sort={\the\glsshortttok},
3875     description={\the\glslongtok},%
3876     first={\protect\glsfirstabbrvfont{\the\glsshortttok}}%
3877     \protect\footnote{\protect\glsfirstlongfont{\the\glslongtok}}},%
3878     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
3879     \protect\footnote{\protect\glsfirstlongfont{\the\glslongpltok}}},%
3880     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.

```

3881 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3882   \glissetattribute{\the\glslabeltok}{nohyperfirst}{true}%
3883   \glshasattribute{\the\glslabeltok}{regular}%
3884   {%
3885     \glissetattribute{\the\glslabeltok}{regular}{false}%
3886   }%
3887   {}%
3888 }%
3889}%
3890{%

```

In case the user wants to mix and match font styles, these are redefined here.

```

3891 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
3892 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvdefaultfont{##1}}%

```

```

3893 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
3894 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

3895 \renewcommand*{\glsxtrfullformat}[2]{%
3896   \glsfirstabbrvfont{\glsaccessshort{##1}}##2%
3897   \protect\footnote{\glsfirstlongfont{\glsaccesslong{##1}}}%
3898 }%
3899 \renewcommand*{\glsxtrfullplformat}[2]{%
3900   \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2%
3901   \protect\footnote{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
3902 }%
3903 \renewcommand*{\Glsxtrfullformat}[2]{%
3904   \glsfirstabbrvfont{\Glsaccessshort{##1}}##2%
3905   \protect\footnote{\glsfirstlongfont{\glsaccesslong{##1}}}%
3906 }%
3907 \renewcommand*{\Glsxtrfullplformat}[2]{%
3908   \glsfirstabbrvfont{\Glsaccessshortpl{##1}}##2%
3909   \protect\footnote{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
3910 }%

```

The first use full form and the inline full form use the short (long) style.

```

3911 \renewcommand*{\glsxtrinlinefullformat}[2]{%
3912   \glsfirstabbrvfont{\glsaccessshort{##1}}##2\glsxtrfullsep{##1}%
3913   (\glsfirstlongfont{\glsaccesslong{##1}})%
3914 }%
3915 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
3916   \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2\glsxtrfullsep{##1}%
3917   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
3918 }%
3919 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
3920   \glsfirstabbrvfont{\Glsaccessshort{##1}}##2\glsxtrfullsep{##1}%
3921   (\glsfirstlongfont{\glsaccesslong{##1}})%
3922 }%
3923 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
3924   \glsfirstabbrvfont{\Glsaccessshortpl{##1}}##2\glsxtrfullsep{##1}%
3925   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
3926 }%
3927 }

```

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. This deferment won't occur with `\glsfirst`.

```

3928 \newabbreviationstyle{postfootnote}%
3929 {%
3930   \renewcommand*{\CustomAbbreviationFields}{%
3931     name={\protect\glsabbrvfont{\the\glsshorttok}},
3932     sort={\the\glsshorttok},
3933     description={\the\glslongtok},%
3934     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%

```

```

3935 \protect\footnote{\protect\glsfirstlongfont{\the\glslongtok}}},%
3936 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
3937 \protect\footnote{\protect\glsfirstlongfont{\the\glslongpltok}}},%
3938 plural={\protect\glsabbvfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

3939 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3940 \csdef{glsxtrpostlink\glscategorylabel}{%
3941 \glsxtrifwasfirstuse{\glsxtrdopostpunc{\protect\footnote
3942 {\glsfirstlongfont{\glsentrylong{\glslabel}}}}}{}%
3943 }%
3944 \glsattribute{\the\glslabeltok}{regular}%
3945 {%
3946 \glssetattribute{\the\glslabeltok}{regular}{false}%
3947 }%
3948 {}}%
3949 }%
3950 }%
3951 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

3952 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
3953 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
3954 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
3955 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

3956 \renewcommand*{\glsxtrfullformat}[2]{%
3957 \glsfirstabbrvfont{\glsaccessshort{##1}}##2%
3958 }%
3959 \renewcommand*{\glsxtrfullplformat}[2]{%
3960 \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2%
3961 }%
3962 \renewcommand*{\Glsxtrfullformat}[2]{%
3963 \glsfirstabbrvfont{\Glsaccessshort{##1}}##2%
3964 }%
3965 \renewcommand*{\Glsxtrfullplformat}[2]{%
3966 \glsfirstabbrvfont{\Glsaccessshortpl{##1}}##2%
3967 }%

```

The first use full form and the inline full form use the short (long) style.

```

3968 \renewcommand*{\glsxtrinlinefullformat}[2]{%
3969 \glsfirstabbrvfont{\glsaccessshort{##1}}##2\glsxtrfullsep{##1}%
3970 (\glsfirstlongfont{\glsaccesslong{##1}})%
3971 }%
3972 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
3973 \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2\glsxtrfullsep{##1}%
3974 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
3975 }%
3976 \renewcommand*{\Glsxtrinlinefullformat}[2]{%

```

```

3977 \glsfirstabbrvfont{\Glsaccessshort{##1}}##2\glxtrfullsep{##1}%
3978 (\glsfirstlongfont{\glsaccesslong{##1}})%
3979 }%
3980 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
3981 \glsfirstabbrvfont{\Glsaccessshortpl{##1}}##2\glxtrfullsep{##1}%
3982 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
3983 }%
3984 }

```

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.

```

3985 \newabbreviationstyle{short}%
3986 {%
3987 \renewcommand*{\CustomAbbreviationFields}{%
3988 name={\protect\glsabbrvfont{\the\glsshorttok}},
3989 sort={\the\glsshorttok},
3990 first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
3991 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
3992 text={\protect\glsabbrvfont{\the\glsshorttok}},
3993 plural={\protect\glsabbrvfont{\the\glsshortpltok}},
3994 description={\the\glslongtok}}%
3995 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3996 \glssetattribute{\the\glslabeltok}{regular}{true}}%
3997 }%
3998 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

3999 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4000 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4001 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4002 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

4003 \renewcommand*{\glxtrinlinefullformat}[2]{%
4004 \protect\glsfirstabbrvfont{\glsaccessshort{##1}}##2\glxtrfullsep{##1}%
4005 (\glsfirstlongfont{\glsaccesslong{##1}})%
4006 }%
4007 \renewcommand*{\glxtrinlinefullplformat}[2]{%
4008 \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}##2\glxtrfullsep{##1}%
4009 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4010 }%
4011 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4012 \protect\glsfirstabbrvfont{\glsaccessshort{##1}}##2\glxtrfullsep{##1}%
4013 (\glsfirstlongfont{\Glsaccesslong{##1}})%
4014 }%
4015 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4016 \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}##2\glxtrfullsep{##1}%
4017 (\glsfirstlongfont{\Glsaccesslongpl{##1}})%

```


4018 }%

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.

```
4019 \renewcommand*{\glxtrfullformat}[2]{%
4020   \glsfirstabbrvfont{\glsaccessshort{##1}}##2%
4021 }%
4022 \renewcommand*{\glxtrfullplformat}[2]{%
4023   \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2%
4024 }%
4025 \renewcommand*{\Glsxtrfullformat}[2]{%
4026   \glsfirstabbrvfont{\glsaccessshort{##1}}##2%
4027 }%
4028 \renewcommand*{\Glsxtrfullplformat}[2]{%
4029   \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2%
4030 }%
4031 }
```

Set this as the default style for acronyms:

```
4032 \setabbreviationstyle[acronym]{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.

```
4033 \newabbreviationstyle{short-desc}%
4034 {%
4035   \renewcommand*{\CustomAbbreviationFields}{%
4036     name={\protect\glsextrinlinefullformat{\the\glslabeltok}},
4037     sort={\the\glsshorttok},
4038     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
4039     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
4040     text={\protect\glsabbrvfont{\the\glsshorttok}},
4041     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
4042     description={\the\glslongtok}}%
4043   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4044     \glssetattribute{\the\glslabeltok}{regular}{true}}%
4045 }%
4046 }
```

In case the user wants to mix and match font styles, these are redefined here.

```
4047 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4048 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4049 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4050 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
```

The inline full form displays the short format followed by the long form in parentheses.

```
4051 \renewcommand*{\glxtrinlinefullformat}[2]{%
4052   \glsfirstabbrvfont{\glsaccessshort{##1}}##2\glxtrfullsep{##1}%
4053   (\glsfirstlongfont{\glsaccesslong{##1}})%
4054 }%
4055 \renewcommand*{\glxtrinlinefullplformat}[2]{%
4056   \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2\glxtrfullplsep{##1}%
4057   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4058 }%
```

```

4056 \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2\glsxtrfullsep{##1}%
4057 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4058 }%
4059 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4060 \glsfirstabbrvfont{\Glsaccessshort{##1}}##2\glsxtrfullsep{##1}%
4061 (\glsfirstlongfont{\glsaccesslong{##1}})%
4062 }%
4063 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4064 \glsfirstabbrvfont{\Glsaccessshortpl{##1}}##2\glsxtrfullsep{##1}%
4065 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4066 }%

```

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.

```

4067 \renewcommand*{\glsxtrfullformat}[2]{%
4068 \glsfirstabbrvfont{\glsaccessshort{##1}}##2%
4069 }%
4070 \renewcommand*{\glsxtrfullplformat}[2]{%
4071 \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2%
4072 }%
4073 \renewcommand*{\Glsxtrfullformat}[2]{%
4074 \glsfirstabbrvfont{\glsaccessshort{##1}}##2%
4075 }%
4076 \renewcommand*{\Glsxtrfullplformat}[2]{%
4077 \glsfirstabbrvfont{\glsaccessshortpl{##1}}##2%
4078 }%
4079 }

```

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.

```

4080 \newabbreviationstyle{long-desc}%
4081 {%
4082 \renewcommand*{\CustomAbbreviationFields}{%
4083 name={\protect\protect\glsfirstlongfont{\the\glslongtok}},
4084 sort={\the\glslongtok},
4085 first={\protect\glsfirstlongfont{\the\glslongtok}},
4086 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
4087 text={\the\glslongtok},
4088 plural={\the\glslongpltok}%
4089 }%
4090 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4091 \glssetattribute{\the\glslabeltok}{regular}{true}}%
4092 }%
4093 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4094 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4095 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4096 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%

```

```
4097 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
```

The inline full form displays the long format followed by the short form in parentheses.

```
4098 \renewcommand*{\glsxtrinlinefullformat}[2]{%
4099   \glsfirstlongfont{\glsaccesslong{##1}}##2\glsxtrfullsep{##1}%
4100   (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
4101 }%
4102 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
4103   \glsfirstlongfont{\glsaccesslongpl{##1}}##2\glsxtrfullsep{##1}%
4104   (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4105 }%
4106 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4107   \glsfirstlongfont{\Glsaccesslong{##1}}##2\glsxtrfullsep{##1}%
4108   (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
4109 }%
4110 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4111   \glsfirstlongfont{\Glsaccesslongpl{##1}}##2\glsxtrfullsep{##1}%
4112   (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4113 }%
```

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.

```
4114 \renewcommand*{\glsxtrfullformat}[2]{%
4115   \glsfirstlongfont{\glsaccesslong{##1}}##2%
4116 }%
4117 \renewcommand*{\glsxtrfullplformat}[2]{%
4118   \glsfirstlongfont{\glsaccesslongpl{##1}}##2%
4119 }%
4120 \renewcommand*{\Glsxtrfullformat}[2]{%
4121   \glsfirstlongfont{\glsaccesslong{##1}}##2%
4122 }%
4123 \renewcommand*{\Glsxtrfullplformat}[2]{%
4124   \glsfirstlongfont{\glsaccesslongpl{##1}}##2%
4125 }%
4126 }
```

`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.

```
4127 \newabbreviationstyle{long}%
4128 {%
4129   \renewcommand*{\CustomAbbreviationFields}{%
4130     name={\protect\glsabbrvfont{\the\glsshorttok}},
4131     sort={\the\glsshorttok},
4132     first={\protect\glsfirstlongfont{\the\glslongtok}},
4133     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
4134     text={\the\glslongtok},
4135     plural={\the\glslongpltok},%
4136     description={\the\glslongtok}%
4137   }%
```

```

4138 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4139 \glssetattribute{\the\glslabeltok}{regular}{true}}%
4140 }%
4141 {%
4142 \GlsXtrUseAbbrStyleFmts{long-desc}%
4143 }

```

1.6.3 Predefined Styles (Small Capitals)

These styles use:

`\glstrscfont`

```

4144 \newcommand*{\glstrscfont}[1]{\textsc{#1}}

```

and for the default short form suffix:

`\glstrscsuffix`

```

4145 \newcommand*{\glstrscsuffix}{\glstextup{\glspluralsuffix}}

```

`long-short-sc`

```

4146 \newabbreviationstyle{long-short-sc}%
4147 {%
4148 \GlsXtrUseAbbrStyleSetup{long-short}%
4149 }%
4150 {%

```

Mostly as long-short style:

```

4151 \GlsXtrUseAbbrStyleFmts{long-short}%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

4152 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
4153 \renewcommand*{\glsabbrvfont}[1]{\glstrscfont{##1}}%
4154 }

```

`g-short-sc-desc`

```

4155 \newabbreviationstyle{long-short-sc-desc}%
4156 {%
4157 \GlsXtrUseAbbrStyleSetup{long-short-desc}%
4158 }%
4159 {%

```

Mostly as long-short-desc style:

```

4160 \GlsXtrUseAbbrStyleFmts{long-short-desc}%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

4161 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
4162 \renewcommand*{\glsabbrvfont}[1]{\glstrscfont{##1}}%
4163 }

```

Now the short (long) version

```
4164 \newabbreviationstyle{short-sc-long}%  
4165 {%  
4166   \GlsXtrUseAbbrStyleSetup{short-long}%  
4167 }%  
4168 {%
```

Mostly as short-long style:

```
4169   \GlsXtrUseAbbrStyleFmts{short-long}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4170   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%  
4171   \renewcommand*\glsabbrvfont[1]{\glstrscfont{##1}}%  
4172 }
```

As before but user provides description

```
4173 \newabbreviationstyle{short-sc-long-desc}%  
4174 {%  
4175   \GlsXtrUseAbbrStyleSetup{short-long-desc}%  
4176 }%  
4177 {%
```

Mostly as short-long-desc style:

```
4178   \GlsXtrUseAbbrStyleFmts{short-long-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4179   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%  
4180   \renewcommand*\glsabbrvfont[1]{\glstrscfont{##1}}%  
4181 }
```

short-sc

```
4182 \newabbreviationstyle{short-sc}%  
4183 {%  
4184   \GlsXtrUseAbbrStyleSetup{short}%  
4185 }%  
4186 {%
```

Mostly as short style:

```
4187   \GlsXtrUseAbbrStyleFmts{short}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4188   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%  
4189   \renewcommand*\glsabbrvfont[1]{\glstrscfont{##1}}%  
4190 }
```

short-sc-desc

```
4191 \newabbreviationstyle{short-sc-desc}%  
4192 {%  
4193   \GlsXtrUseAbbrStyleSetup{short-desc}%  
4194 }%  
4195 {%
```

Mostly as short style:

```
4196 \GlsXtrUseAbbrStyleFmts{short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4197 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
```

```
4198 \renewcommand*\glsabbrvfont[1]{\glstrscfont{##1}}%
```

```
4199 }
```

long-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
4200 \newabbreviationstyle{long-sc}%
```

```
4201 {%
```

```
4202 \GlsXtrUseAbbrStyleSetup{long}%
```

```
4203 }%
```

```
4204 {%
```

Mostly as long style:

```
4205 \GlsXtrUseAbbrStyleFmts{long}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4206 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
```

```
4207 \renewcommand*\glsabbrvfont[1]{\glstrscfont{##1}}%
```

```
4208 }
```

long-desc-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
4209 \newabbreviationstyle{long-desc-sc}%
```

```
4210 {%
```

```
4211 \GlsXtrUseAbbrStyleSetup{long-desc}%
```

```
4212 }%
```

```
4213 {%
```

Mostly as long style:

```
4214 \GlsXtrUseAbbrStyleFmts{long-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4215 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
```

```
4216 \renewcommand*\glsabbrvfont[1]{\glstrscfont{##1}}%
```

```
4217 }
```

footnote-sc

```
4218 \newabbreviationstyle{footnote-sc}%
```

```
4219 {%
```

```
4220 \GlsXtrUseAbbrStyleSetup{footnote}%
```

```
4221 }%
```

```
4222 {%
```

Mostly as long style:

```
4223 \GlsXtrUseAbbrStyleFmts{footnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4224 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4225 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
4226 }
```

postfootnote-sc

```
4227 \newabbreviationstyle{postfootnote-sc}%
4228 {%
4229 \GlsXtrUseAbbrStyleSetup{postfootnote}%
4230 }%
4231 {%
```

Mostly as long style:

```
4232 \GlsXtrUseAbbrStyleFmts{postfootnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4233 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4234 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
4235 }
```

1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the relsize package, which must be loaded by the user. These styles all use:

\glxtrsmfont

```
4236 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}
```

and for the default short form suffix:

\glxtrsmsuffix

```
4237 \newcommand*{\glxtrsmsuffix}{\glspluralsuffix}
```

long-short-sm

```
4238 \newabbreviationstyle{long-short-sm}%
4239 {%
4240 \GlsXtrUseAbbrStyleSetup{long-short}%
4241 }%
4242 {%
```

Mostly as long-short style:

```
4243 \GlsXtrUseAbbrStyleFmts{long-short}%
4244 \renewcommand*{\glssabrvfont[1]{\glxtrsmfont{##1}}}%
4245 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
4246 }
```

g-short-sm-desc

```
4247 \newabbreviationstyle{long-short-sm-desc}%
4248 {%
4249 \GlsXtrUseAbbrStyleSetup{long-short-desc}%
```

4250 }%

4251 {%

Mostly as long-short-desc style:

4252 \GlsXtrUseAbbrStyleFmts{long-short-desc}%

4253 \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%

4254 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%

4255 }

short-sm-long Now the short (long) version

4256 \newabbreviationstyle{short-sm-long}%

4257 {%

4258 \GlsXtrUseAbbrStyleSetup{short-long}%

4259 }%

4260 {%

Mostly as short-long style:

4261 \GlsXtrUseAbbrStyleFmts{short-long}%

4262 \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%

4263 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%

4264 }

rt-sm-long-desc As before but user provides description

4265 \newabbreviationstyle{short-sm-long-desc}%

4266 {%

4267 \GlsXtrUseAbbrStyleSetup{short-long-desc}%

4268 }%

4269 {%

Mostly as short-long-desc style:

4270 \GlsXtrUseAbbrStyleFmts{short-long-desc}%

4271 \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%

4272 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%

4273 }

short-sm

4274 \newabbreviationstyle{short-sm}%

4275 {%

4276 \GlsXtrUseAbbrStyleSetup{short}%

4277 }%

4278 {%

Mostly as short style:

4279 \GlsXtrUseAbbrStyleFmts{short}%

4280 \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%

4281 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%

4282 }

short-sm-desc

4283 \newabbreviationstyle{short-sm-desc}%


```

4284 {%
4285   \GlsXtrUseAbbrStyleSetup{short-desc}%
4286 }%
4287 {%
  Mostly as short style:
4288   \GlsXtrUseAbbrStyleFmts{short-desc}%
4289   \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%
4290   \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4291 }

```

long-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like \glsshort.

```

4292 \newabbreviationstyle{long-sm}%
4293 {%
4294   \GlsXtrUseAbbrStyleSetup{long}%
4295 }%
4296 {%
  Mostly as long style:
4297   \GlsXtrUseAbbrStyleFmts{long}%
4298   \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%
4299   \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4300 }

```

long-desc-sm The smaller font will only be used if the short form is explicitly invoked through commands like \glsshort.

```

4301 \newabbreviationstyle{long-desc-sm}%
4302 {%
4303   \GlsXtrUseAbbrStyleSetup{long-desc}%
4304 }%
4305 {%
  Mostly as long style:
4306   \GlsXtrUseAbbrStyleFmts{long-desc}%
4307   \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%
4308   \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4309 }

```

footnote-sm

```

4310 \newabbreviationstyle{footnote-sm}%
4311 {%
4312   \GlsXtrUseAbbrStyleSetup{footnote}%
4313 }%
4314 {%
  Mostly as long style:
4315   \GlsXtrUseAbbrStyleFmts{footnote}%
4316   \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%
4317   \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4318 }

```

postfootnote-sm

```
4319 \newabbreviationstyle{postfootnote-sm}%
4320 {%
4321   \GlsXtrUseAbbrStyleSetup{postfootnote}%
4322 }%
4323 {%
```

Mostly as long style:

```
4324   \GlsXtrUseAbbrStyleFmts{postfootnote}%
4325   \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
4326   \renewcommand*\{abbrvpluralsuffix}{\protect\glstrsmsuffix}%
4327 }
```

1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

long-short-em

```
4328 \newabbreviationstyle{long-short-em}%
4329 {%
4330   \GlsXtrUseAbbrStyleSetup{long-short}%
4331 }%
4332 {%
```

Mostly as long-short style:

```
4333   \GlsXtrUseAbbrStyleFmts{long-short}%
4334   \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4335 }
```

g-short-em-desc

```
4336 \newabbreviationstyle{long-short-em-desc}%
4337 {%
4338   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
4339 }%
4340 {%
```

Mostly as long-short-desc style:

```
4341   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
4342   \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4343 }
```

short-em-long Now the short (long) version

```
4344 \newabbreviationstyle{short-em-long}%
4345 {%
4346   \GlsXtrUseAbbrStyleSetup{short-long}%
4347 }%
4348 {%
```

Mostly as short-long style:

```
4349 \GlsXtrUseAbbrStyleFmts{short-long}%
4350 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4351 }
```

rt-em-long-desc As before but user provides description

```
4352 \newabbreviationstyle{short-em-long-desc}%
4353 {%
4354 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
4355 }%
4356 {%
```

Mostly as short-long-desc style:

```
4357 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
4358 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4359 }
```

short-em

```
4360 \newabbreviationstyle{short-em}%
4361 {%
4362 \GlsXtrUseAbbrStyleSetup{short}%
4363 }%
4364 {%
```

Mostly as short style:

```
4365 \GlsXtrUseAbbrStyleFmts{short}%
4366 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4367 }
```

short-em-desc

```
4368 \newabbreviationstyle{short-em-desc}%
4369 {%
4370 \GlsXtrUseAbbrStyleSetup{short-desc}%
4371 }%
4372 {%
```

Mostly as short style:

```
4373 \GlsXtrUseAbbrStyleFmts{short-desc}%
4374 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4375 }
```

long-em The emphasized font will only be used if the short form is explicitly invoked through commands like \glsshort.

```
4376 \newabbreviationstyle{long-em}%
4377 {%
4378 \GlsXtrUseAbbrStyleSetup{long}%
4379 }%
4380 {%
```

Mostly as long style:

```
4381 \GlsXtrUseAbbrStyleFmts{long}%
4382 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4383 }
```

long-desc-em The emphasized font will only be used if the short form is explicitly invoked through commands like \glssshort.

```
4384 \newabbreviationstyle{long-desc-em}%
4385 {%
4386 \GlsXtrUseAbbrStyleSetup{long-desc}%
4387 }%
4388 {%
```

Mostly as long style:

```
4389 \GlsXtrUseAbbrStyleFmts{long-desc}%
4390 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4391 }
```

footnote-em

```
4392 \newabbreviationstyle{footnote-em}%
4393 {%
4394 \GlsXtrUseAbbrStyleSetup{footnote}%
4395 }%
4396 {%
```

Mostly as long style:

```
4397 \GlsXtrUseAbbrStyleFmts{footnote}%
4398 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4399 }
```

postfootnote-em

```
4400 \newabbreviationstyle{postfootnote-em}%
4401 {%
4402 \GlsXtrUseAbbrStyleSetup{postfootnote}%
4403 }%
4404 {%
```

Mostly as long style:

```
4405 \GlsXtrUseAbbrStyleFmts{postfootnote}%
4406 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4407 }
```

1.7 Using Entries in Headings

There are four main problems with using entries in sectioning commands: they can mess with the first use flag if they end up in the table of contents, they can add unwanted numbers to the entry's location list, the label is corrupted if used inside \MakeUppercase (which is used by the default headings style) and they need to be expandable for PDF bookmarks.

The glossaries package therefore recommends the use of the expandable commands, such as `\glsentryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glsabbrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref's \texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glsxtrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to `false`, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that `glossaries` automatically loads `textcase`, so the label can be protected from case change with `textcase's \NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
4408 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
4409 \renewcommand*{\markright}[1]{%
4410   \glsxtrmarkhook
4411   \@glsxtr@org@markright{#1}%
4412   \glsxtrrestoremarkhook
4413 }
```

`\markboth` Save original definition:

```
4414 \let\@glsxtr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
4415 \renewcommand*{\markboth}[2]{%
4416   \glsxtrmarkhook
4417   \@glsxtr@org@markboth{#1}{#2}%
4418   \glsxtrrestoremarkhook
4419 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

`sxtrRevertMarks`

```
4420 \newcommand*{\glsxtrRevertMarks}{%
4421   \let\markright\@glsxtr@org@markright
4422   \let\markboth\@glsxtr@org@markboth
4423 }
```

`\glsxtrmarkhook` Hook used in new definition of `\markboth` and `\markright` to make some changes to apply to the marks:

```
4424 \newcommand*{\glsxtrmarkhook}{%
```

Save current definitions:

```

4425 \let\@glxstr@org@MakeUppercase\MakeUppercase
4426 \let\@glxstr@org@glxstrtitleshort\glxstrtitleshort
4427 \let\@glxstr@org@glxstrtitleshortpl\glxstrtitleshortpl
4428 \let\@glxstr@org@Glsxtrtitleshort\Glsxtrtitleshort
4429 \let\@glxstr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
4430 \let\@glxstr@org@glxstrtitletext\glxstrtitletext
4431 \let\@glxstr@org@Glsxtrtitletext\Glsxtrtitletext
4432 \let\@glxstr@org@glxstrtitleplural\glxstrtitleplural
4433 \let\@glxstr@org@Glsxtrtitleplural\Glsxtrtitleplural
4434 \let\@glxstr@org@glxstrtitlefirst\glxstrtitlefirst
4435 \let\@glxstr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
4436 \let\@glxstr@org@glxstrtitlefirstplural\glxstrtitlefirstplural
4437 \let\@glxstr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural

```

New definitions

```

4438 \let\MakeUppercase\MakeTextUppercase
4439 \let\glxstrtitleshort\glxstrheadshort
4440 \let\glxstrtitleshortpl\glxstrheadshortpl
4441 \let\Glsxtrtitleshort\Glsxtrheadshort
4442 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
4443 \let\glxstrtitletext\glxstrheadtext
4444 \let\Glsxtrtitletext\Glsxtrheadtext
4445 \let\glxstrtitleplural\glxstrheadplural
4446 \let\Glsxtrtitleplural\Glsxtrheadplural
4447 \let\glxstrtitlefirst\glxstrheadfirst
4448 \let\Glsxtrtitlefirst\Glsxtrheadfirst
4449 \let\glxstrtitlefirstplural\glxstrheadfirstplural
4450 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
4451 }

```

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.)

```

4452 \newcommand*{\glxstrrestoremarkhook}{%
4453 \let\MakeUppercase\@glxstr@org@MakeUppercase
4454 \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
4455 \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
4456 \let\Glsxtrtitleshort\@glxstr@org@Glsxtrtitleshort
4457 \let\Glsxtrtitleshortpl\@glxstr@org@Glsxtrtitleshortpl
4458 \let\glxstrtitletext\@glxstr@org@glxstrtitletext
4459 \let\Glsxtrtitletext\@glxstr@org@Glsxtrtitletext
4460 \let\glxstrtitleplural\@glxstr@org@glxstrtitleplural
4461 \let\Glsxtrtitleplural\@glxstr@org@Glsxtrtitleplural
4462 \let\glxstrtitlefirst\@glxstr@org@glxstrtitlefirst
4463 \let\Glsxtrtitlefirst\@glxstr@org@Glsxtrtitlefirst
4464 \let\glxstrtitlefirstplural\@glxstr@org@glxstrtitlefirstplural
4465 \let\Glsxtrtitlefirstplural\@glxstr@org@Glsxtrtitlefirstplural

```

4466 }

Instead of using one document-wide conditional, use headuc attribute to determine whether or not to use the all upper case form.

glsxtrheadshort Command used to display short form in the page header.

```
4467 \newcommand*{\glsxtrheadshort}[1]{%
4468   \protect\NoCaseChange
4469   {%
4470     \glsifattribute{#1}{headuc}{true}%
4471     {%
4472       \GLSxtrshort[noindex,hyper=false]{#1}[]%
4473     }%
4474     {%
4475       \glsxtrshort[noindex,hyper=false]{#1}[]%
4476     }%
4477   }%
4478 }
```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```
4479 \newrobustcmd*{\lsxtrtitleshort}[1]{%
4480   \glsxtrshort[noindex,hyper=false]{#1}[]%
4481 }
```

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.

```
4482 \newcommand*{\glsxtrheadshortpl}[1]{%
4483   \protect\NoCaseChange
4484   {%
4485     \glsifattribute{#1}{headuc}{true}%
4486     {%
4487       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
4488     }%
4489     {%
4490       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
4491     }%
4492   }%
4493 }
```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```
4494 \newrobustcmd*{\lsxtrtitleshortpl}[1]{%
4495   \glsxtrshort[noindex,hyper=false]{#1}[]%
4496 }
```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```
4497 \newcommand*{\Glsxtrheadshort}[1]{%
```

```

4498 \protect\NoCaseChange
4499 {%
4500   \glsifattribute{#1}{headuc}{true}%
4501   {%
4502     \GLSxtrshort[noindex,hyper=false]{#1}[]%
4503   }%
4504   {%
4505     \GLsxtrshort[noindex,hyper=false]{#1}[]%
4506   }%
4507 }%
4508 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

4509 \newrobustcmd*{\GLsxtrtitleshort}[1]{%
4510   \GLSxtrshort[noindex,hyper=false]{#1}[]%
4511 }

```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```

4512 \newcommand*{\GLsxtrheadshortpl}[1]{%
4513   \protect\NoCaseChange
4514   {%
4515     \glsifattribute{#1}{headuc}{true}%
4516     {%
4517       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
4518     }%
4519     {%
4520       \GLsxtrshortpl[noindex,hyper=false]{#1}[]%
4521     }%
4522   }%
4523 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

4524 \newrobustcmd*{\GLsxtrtitleshortpl}[1]{%
4525   \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
4526 }

```

\glsxtrheadtext As above but for the text value.

```

4527 \newcommand*{\glsxtrheadtext}[1]{%
4528   \protect\NoCaseChange
4529   {%
4530     \glsifattribute{#1}{headuc}{true}%
4531     {%
4532       \GLStext[noindex,hyper=false]{#1}[]%
4533     }%
4534   }%

```



```

4535     \glstext[noindex,hyper=false]{#1}[]%
4536   }%
4537 }%
4538 }

```

glsxtrtitletext Command to display text value in section title and table of contents.

```

4539 \newrobustcmd*{\glsxtrtitletext}[1]{%
4540   \glstext[noindex,hyper=false]{#1}[]%
4541 }

```

\Glsxtrheadtext First letter converted to upper case

```

4542 \newcommand*{\Glsxtrheadtext}[1]{%
4543   \protect\NoCaseChange
4544   {%
4545     \glsifattribute{#1}{headuc}{true}%
4546     {%
4547       \GLStext[noindex,hyper=false]{#1}[]%
4548     }%
4549     {%
4550       \Glstext[noindex,hyper=false]{#1}[]%
4551     }%
4552   }%
4553 }

```

Glsxtrtitletext Command to display text value in section title and table of contents with the first letter changed to upper case.

```

4554 \newrobustcmd*{\Glsxtrtitletext}[1]{%
4555   \GLstext[noindex,hyper=false]{#1}[]%
4556 }

```

lgsxtrheadplural As above but for the plural value.

```

4557 \newcommand*{\lgsxtrheadplural}[1]{%
4558   \protect\NoCaseChange
4559   {%
4560     \glsifattribute{#1}{headuc}{true}%
4561     {%
4562       \GLSplural[noindex,hyper=false]{#1}[]%
4563     }%
4564     {%
4565       \glsplural[noindex,hyper=false]{#1}[]%
4566     }%
4567   }%
4568 }

```

gsxtrtitleplural Command to display plural value in section title and table of contents.

```

4569 \newrobustcmd*{\gsxtrtitleplural}[1]{%
4570   \glsplural[noindex,hyper=false]{#1}[]%
4571 }

```

`lsxtrheadplural` Convert first letter to upper case.

```
4572 \newcommand*{\Glsxtrheadplural}[1]{%
4573   \protect\NoCaseChange
4574   {%
4575     \glsifattribute{#1}{headuc}{true}%
4576     {%
4577       \GLSplural[noindex,hyper=false]{#1}[]%
4578     }%
4579     {%
4580       \Glsplural[noindex,hyper=false]{#1}[]%
4581     }%
4582   }%
4583 }
```

`sxtrtitleplural` Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
4584 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
4585   \Glsplural[noindex,hyper=false]{#1}[]%
4586 }
```

`glxtrheadfirst` As above but for the first value.

```
4587 \newcommand*{\glxtrheadfirst}[1]{%
4588   \protect\NoCaseChange
4589   {%
4590     \glsifattribute{#1}{headuc}{true}%
4591     {%
4592       \GLSfirst[noindex,hyper=false]{#1}[]%
4593     }%
4594     {%
4595       \glsfirst[noindex,hyper=false]{#1}[]%
4596     }%
4597   }%
4598 }
```

`lsxtrtitlefirst` Command to display first value in section title and table of contents.

```
4599 \newrobustcmd*{\glxtrtitlefirst}[1]{%
4600   \glsfirst[noindex,hyper=false]{#1}[]%
4601 }
```

`Glsxtrheadfirst` First letter converted to upper case

```
4602 \newcommand*{\Glsxtrheadfirst}[1]{%
4603   \protect\NoCaseChange
4604   {%
4605     \glsifattribute{#1}{headuc}{true}%
4606     {%
4607       \GLSfirst[noindex,hyper=false]{#1}[]%
4608     }%
4609     {%
```

```

4610     \Glsfirst[noindex,hyper=false]{#1}[]%
4611 }%
4612 }%
4613 }

\lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter
                  changed to upper case.
4614 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
4615   \Glsfirst[noindex,hyper=false]{#1}[]%
4616 }

\headfirstplural As above but for the firstplural value.
4617 \newcommand*{\Glsxtrheadfirstplural}[1]{%
4618   \protect\NoCaseChange
4619   {%
4620     \glsifattribute{#1}{headuc}{true}%
4621     {%
4622       \GLSfirstplural[noindex,hyper=false]{#1}[]%
4623     }%
4624     {%
4625       \glsfirstplural[noindex,hyper=false]{#1}[]%
4626     }%
4627   }%
4628 }

\itlefirstplural Command to display firstplural value in section title and table of contents.
4629 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
4630   \glsfirstplural[noindex,hyper=false]{#1}[]%
4631 }

\headfirstplural First letter converted to upper case
4632 \newcommand*{\Glsxtrheadfirstplural}[1]{%
4633   \protect\NoCaseChange
4634   {%
4635     \glsifattribute{#1}{headuc}{true}%
4636     {%
4637       \GLSfirstplural[noindex,hyper=false]{#1}[]%
4638     }%
4639     {%
4640       \Glsfirstplural[noindex,hyper=false]{#1}[]%
4641     }%
4642   }%
4643 }

\itlefirstplural Command to display first value in section title and table of contents with the first letter
                  changed to upper case.
4644 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
4645   \Glsfirstplural[noindex,hyper=false]{#1}[]%
4646 }

```

`\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.

```
4647 \ifdef\texorpdfstring
4648 {
4649   \newcommand*\glsfmtshort[1]{%
4650     \texorpdfstring
4651       {\glsxtrtitleshort{#1}}%
4652       {\glsentryshort{#1}}%
4653   }
4654 }
4655 {
4656   \newcommand*\glsfmtshort[1]{%
4657     \glsxtrtitleshort{#1}}
4658 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
4659 \ifdef\texorpdfstring
4660 {
4661   \newcommand*\glsfmtshortpl[1]{%
4662     \texorpdfstring
4663       {\glsxtrtitleshortpl{#1}}%
4664       {\glsentryshortpl{#1}}%
4665   }
4666 }
4667 {
4668   \newcommand*\glsfmtshortpl[1]{%
4669     \glsxtrtitleshortpl{#1}}
4670 }
```

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).

```
4671 \ifdef\texorpdfstring
4672 {
4673   \newcommand*\Glsfmtshort[1]{%
4674     \texorpdfstring
4675       {\Glsxtrtitleshort{#1}}%
4676       {\glsentryshort{#1}}%
4677   }
4678 }
4679 {
4680   \newcommand*\Glsfmtshort[1]{%
4681     \Glsxtrtitleshort{#1}}
4682 }
```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

4683 \ifdef\texorpdfstring
4684 {
4685   \newcommand*\Glsfmtshortpl}[1]{%
4686     \texorpdfstring
4687     {\Glsxtrtitleshortpl{#1}}%
4688     {\glsentryshortpl{#1}}%
4689   }
4690 }
4691 {
4692   \newcommand*\Glsfmtshortpl}[1]{%
4693     \Glsxtrtitleshortpl{#1}}
4694 }

```

`\glsfmttext` As above but for the text value.

```

4695 \ifdef\texorpdfstring
4696 {
4697   \newcommand*\glsfmttext}[1]{%
4698     \texorpdfstring
4699     {\glsxtrtitletext{#1}}%
4700     {\glsentrytext{#1}}%
4701   }
4702 }
4703 {
4704   \newcommand*\glsfmttext}[1]{%
4705     \glsxtrtitletext{#1}}
4706 }

```

`\Glsfmttext` First letter converted to upper case.

```

4707 \ifdef\texorpdfstring
4708 {
4709   \newcommand*\Glsfmttext}[1]{%
4710     \texorpdfstring
4711     {\Glsxtrtitletext{#1}}%
4712     {\glsentrytext{#1}}%
4713   }
4714 }
4715 {
4716   \newcommand*\Glsfmttext}[1]{%
4717     \Glsxtrtitletext{#1}}
4718 }

```

`\glsfmtplural` As above but for the plural value.

```

4719 \ifdef\texorpdfstring
4720 {
4721   \newcommand*\glsfmtplural}[1]{%
4722     \texorpdfstring
4723     {\glsxtrtitleplural{#1}}%
4724     {\glsentryplural{#1}}%
4725   }

```

```

4726 }
4727 {
4728   \newcommand*{\glsfmtplural}[1]{%
4729     \glsxtrtitleplural{#1}}
4730 }

```

`\Glsfmtplural` First letter converted to upper case.

```

4731 \ifdef\texorpdfstring
4732 {
4733   \newcommand*{\Glsfmtplural}[1]{%
4734     \texorpdfstring
4735     {\Glsxtrtitleplural{#1}}%
4736     {\glsentryplural{#1}}%
4737   }
4738 }
4739 {
4740   \newcommand*{\Glsfmtplural}[1]{%
4741     \Glsxtrtitleplural{#1}}
4742 }

```

`\glsfmtfirst` As above but for the first value.

```

4743 \ifdef\texorpdfstring
4744 {
4745   \newcommand*{\glsfmtfirst}[1]{%
4746     \texorpdfstring
4747     {\glsxtrtitlefirst{#1}}%
4748     {\glsentryfirst{#1}}%
4749   }
4750 }
4751 {
4752   \newcommand*{\glsfmtfirst}[1]{%
4753     \glsxtrtitlefirst{#1}}
4754 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

4755 \ifdef\texorpdfstring
4756 {
4757   \newcommand*{\Glsfmtfirst}[1]{%
4758     \texorpdfstring
4759     {\Glsxtrtitlefirst{#1}}%
4760     {\glsentryfirst{#1}}%
4761   }
4762 }
4763 {
4764   \newcommand*{\Glsfmtfirst}[1]{%
4765     \Glsxtrtitlefirst{#1}}
4766 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

4767 \ifdef\teorpdfstring
4768 {
4769   \newcommand*{\glsfmtfirstpl}[1]{%
4770     \teorpdfstring
4771     {\glsxtrtitlefirstplural{#1}}%
4772     {\glsentryfirstplural{#1}}%
4773   }
4774 }
4775 {
4776   \newcommand*{\glsfmtfirstpl}[1]{%
4777     \glsxtrtitlefirstplural{#1}}
4778 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

4779 \ifdef\teorpdfstring
4780 {
4781   \newcommand*{\Glsfmtfirstpl}[1]{%
4782     \teorpdfstring
4783     {\Glsxtrtitlefirstplural{#1}}%
4784     {\glsentryfirstplural{#1}}%
4785   }
4786 }
4787 {
4788   \newcommand*{\Glsfmtfirstpl}[1]{%
4789     \Glsxtrtitlefirstplural{#1}}
4790 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

4791 \newcommand*{\RequireGlossariesExtraLang}[1]{%
4792   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
4793 }

```

`sariesExtraLang`

```

4794 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
4795   \ProvidesFile{glossariesxtr-#1.ldf}%
4796 }

```

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.)

```

4797 \@ifpackageloaded{tracklang}
4798 {%

```

```

4799 \AnyTrackedLanguages
4800 {%
4801   \ForEachTrackedDialect{\this@dialect}{%
4802     \IfTrackedLanguageFileExists{\this@dialect}%
4803     {glossariesxtr-}% prefix
4804     {.ldf}%
4805     {%
4806       \RequireGlossariesExtraLang{\CurrentTrackedTag}%
4807     }%
4808     {%
4809     }%
4810   }%
4811 }%
4812 {}%
4813 }
4814 {}

```


Glossary

This document is incomplete. The external file associated with the glossary ‘main’ (which should be called `glossaries-extra-code.gls2`) hasn’t been created.

Check the contents of the file `glossaries-extra-code.gls2`. If it’s empty, that means you haven’t indexed any of your entries in this glossary (using commands like `\gls` or `\glsadd`) so this list can’t be generated. If the file isn’t empty, the document build process hasn’t been completed.

If you don’t want this glossary, add `nomain` to your package option list when you load `glossaries-extra.sty`. For example:

```
\usepackage[nomain]{glossaries-extra}
```

Try one of the following:

- Add `automake` to your package option list when you load `glossaries-extra.sty`. For example:

```
\usepackage[automake]{glossaries-extra}
```

- Run the external (Lua) application:

```
makeglossaries-lite "glossaries-extra-code"
```

- Run the external (Perl) application:

```
makeglossaries "glossaries-extra-code"
```

Then rerun \LaTeX on this document.

This message will be removed once the problem has been fixed.

Change History

0.1 (2015-11-22)		
General: Initial experimental release	4
0.2 (2015-11-30)		
\Glsfmtshort: new	140
\glsfmtshort: new	140
\Glsfmtshortpl: new	140
\glsfmtshortpl: new	140
short: switched inline full form to short		
(long)	120
0.3 (2015-12-02)		
\@ACRlong: added redefinition	28
\@ACRlongpl: added redefinition	29
\@ACRshort: added redefinition	26
\@ACRshortpl: added redefinition	27
\@Acrlong: added redefinition	27
\@Acrlongpl: added redefinition	28
\@Acrshort: added redefinition	25
\@Acrshortpl: added redefinition	26
\@GLSdesc: added redefinition	22
\@GLSdescplural@: added redefinition	.	22
\@GLSfirst@: added redefinition	20
\@GLSname@: added redefinition	22
\@GLSplural: added redefinition	21
\@GLSSymbol@: added redefinition	23
\@GLSSymbolplural@: added redefini-		
tion	23
\@GLStext@: added redefinition	20
\@GLSuseri@: added redefinition	23
\@GLSuserii@: added redefinition	24
\@GLSuseriii@: added redefinition	24
\@GLSuseriv@: added redefinition	24
\@GLSuseriv@: added redefinition	24
\@GLSuservi@: added redefinition	25
\@acrlong: added redefinition	27
\@acrlongpl: added redefinition	28
\@acrshort: added redefinition	25
\@acrshortpl: added redefinition	26
\@gls@field@link: added optional ar-		
gument	19
\@glsdescplural@: added redefinition	.	22
\@glsfirst@: added redefinition	20
\@glsfirstplural@: added redefinition		21
\@glsplural: added redefinition	20
\@glssymbolplural@: added redefini-		
tion	23
\@glxtr@defaultnoglossarywarning:		
new	61
\@glxtr@field@linkdefs: new	19
\@glxtr@insertdots: new	96
\@print@glossary: added redefinition	.	58
\glsabbrvdefaultfont: renamed from		
\abbrvdefaultfont	100
\glsaccessdesc: new	69
\glsaccessdescplural: new	70
\glsaccessfirst: new	65
\glsaccessfirstplural: new	66
\glsaccesslong: new	72
\glsaccesslong: new	72
\glsaccessname: new	63
\glsaccessplural: new	65
\glsaccessshort: new	71
\glsaccessshort: new	71
\glsaccessshortpl: new	72
\glsaccessshortpl: new	71

\glsaccesssymbol: new	67	\@cGLSpl@: new	44
\glsaccesssymbolplural: new	68	\@glstr@setentrycountunsetattr:	
\glsaccesstext: new	64	new	39
\glstentryfmt: added check for short ..	19	\cGLS: new	44
\glslongpltok: new	96	\cGLSformat: new	44
\glsshortpltok: new	96	\cGLSpl: new	44
\glstrdiscardperiod: added check		\cGLSplformat: new	45
for plural	94	\GlossariesExtraWarningNoLine:	
\GLSxtrlongpl: new	109	new	6
\Glsxtrlongpl: new	108	\glsenableentrycount: new	40
\glxtrlongpl: new	108	\glsfirstabbrvdefaultfont: new ..	100
\glxtrNoGlossaryWarning: new	9	\glsfirstlongdefaultfont: new ...	100
\glxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirst: new	142
new	93	\Glsfmtfirst: new	142
\glxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtfirstpl: new	143
new	93	\Glsfmtfirstpl: new	142
\glxtrpostlinkendsentence: new ..	93	\Glsfmtplural: new	142
\GLSxtrshortpl: new	107	\Glsfmtplural: new	141
\Glsxtrshortpl: new	107	\Glsfmtshort: changed to use	
\glxtrshortpl: new	106	\Glsxtrtitleshort	140
short-long-desc: fixed name to use		renamed from \Glstentryfmtshort .	140
\glslabeltok	116	\Glsfmtshort: changed to use	
\newabbreviation: fixed family name in		\glxtrtitleshort	140
\setkeys	97	renamed from \glstentryfmtshort .	140
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	115	\Glsxtrtitleshortpl	140
0.4 (2015-12-03)		renamed from \Glstentryfmtshortpl	140
\@glstr@doabbreviationsdef: added		\Glsfmtshortpl: changed to use	
redefinition of \acronymtype	6	\glxtrtitleshortpl	140
\Glsfmtshort: changed to use		renamed from \glstentryfmtshortpl	140
\Glsxtrshort	140	\Glsfmttext: new	141
\Glsfmtshort: changed to use		\Glsfmttext: new	141
\glxtrshort	140	\glshasattribute: new	77
\Glsfmtshortpl: changed to use		\glshascategoryattribute: new	77
\glxtrshortpl	140	\GlsXtrEnableEntryCounting: new ..	39
\Glsfmtshortpl: changed to use		\glxtrifcounttrigger: new	42
\glxtrshortpl	140	\glxtrscfont: new	124
\glxtrifemptyglossary: new	10	\glxtrscsuffix: new	124
\glxtrnewnumber: added extra argu-		\glxtrsmfont: new	127
ment	80	\glxtrsmsuffix: new	127
\glxtrnewsymbol: added extra argu-		short-em: new	131
ment	80	short-em-desc: new	131
\MakeAcronymsAbbreviations: set the		short-em-long: new	130
default type to \acronymtype	53	short-em-long-desc: new	131
\newterm: fixed name argument	80	short-sm: new	128
0.5 (2015-12-07)		short-sm-desc: new	128
\@cGLS: new	44	short-sm-long: new	128
\@cGLS@: new	44	short-sm-long-desc: new	128
\@cGLSpl: new	44	long-desc-em: new	132

long-desc-sm: new	129	\glxtrheadtext: now uses headuc at-tribute	136
long-em: new	131	short-long: switch off regular attribute if set	116
long-short-em: new	130	short-long-desc: switch off regular at-tribute if set	117
long-short-em-desc: new	130	long-short: switch off regular attribute if set	114
long-short-sm: new	127	long-short-desc: switch off regular at-tribute if set	115
long-short-sm-desc: new	127	footnote: switch off regular attribute if set	117
long-sm: new	129	postfootnote: switch off regular at-tribute if set	119
footnote-em: new	132		
footnote-sc: new	126	0.5.2 (2015-12-08)	
footnote-sm: new	129	\@GLSdesc@: added	70
postfootnote-em: new	132	\@GLSdescplural@: added	71
postfootnote-sc: new	127	\@GLSfirst@: added	66
postfootnote-sm: new	130	\@GLSfirstplural@: added	67
0.5.1 (2015-12-02)		\@GLSname@: added	64
\Glsaccessstext: new	64	\@GLSplural@: added	65
0.5.1 (2015-12-07)		\@GLSsymbol@: added	68
\@glxtr@doaccsupp: new	9	\@GLSsymbolplural@: added	69
General: removed \ifglxtruseuchead	135	\@GLStext@: added	64
\Glsaccessdesc: new	69	\@GLSdesc@: added	70
\Glsaccessdescplural: new	70	\@GLSdescplural@: added	70
\Glsaccessfirst: new	66	\@Glsfirst@: added	66
\Glsaccessfirstplural: new	67	\@Glsfirstplural@: added	67
\Glsaccessname: new	63	\@Glsname@: added	63
\Glsaccessplural: new	65	\@Glsplural@: added	65
\Glsaccesssymbol: new	68	\@Glsymbol@: added	68
\Glsaccesssymbolplural: new	68	\@Glsymbolplural@: added	69
\Glsxtrheadfirst: now uses headuc at-tribute	138	\@Glstext@: added	64
\glxtrheadfirst: now uses headuc at-tribute	138	\@glSdesc@: added	69
\Glsxtrheadfirstplural: now uses headuc attribute	139	\@glSdescplural@: added	70
\glxtrheadfirstplural: now uses headuc attribute	139	\@glSfirst@: added	66
\Glsxtrheadplural: now uses headuc attribute	138	\@glSfirstplural@: added	67
\glxtrheadplural: now uses headuc attribute	137	\@glSname@: added	63
\Glsxtrheadshort: now uses headuc at-tribute	135	\@glSplural@: added	65
\glxtrheadshort: now uses headuc at-tribute	135	\@glSsymbol@: added	67
\Glsxtrheadshortpl: now uses headuc attribute	136	\@glSsymbolplural@: added	68
\glxtrheadshortpl: now uses headuc attribute	135	\@glStext@: added	64
\Glsxtrheadtext: now uses headuc at-tribute	137	\@glxtr@activate@initialtagging: new	92
		\@glxtr@do@titlecaps@warn: new ..	91
		\@glxtr@tag: new	92
		General: fixed typo in glossaries-accsupp and tidied up code to use just one \@ifpackageloaded	63

removed \glsxtrabbrvfmt	109	0.5.4 ()	
\glossaryentrynumbers:added	18	\@glsxtr@setentryunitcountunsetattr:	
\Glossentrydesc:added	89	new	51
\Glossentryname:added	84	0.5.4 (2015-12-15)	
\Glossentrysymbol:added	90	\@newglossaryentry@defunitcounters:	
\glossentrysymbol:added	90	new	45
\GLSaccessdesc:new	70, 75	\@GLSxtr@p@acrlong@:new	38
\GLSaccessdescplural:new	71, 75	\@GLSxtr@p@acrlongpl@:new	38
\GLSaccessfirst:new	66, 74	\@GLSxtr@p@acrshort@:new	37
\GLSaccessfirstplural:new	67, 74	\@GLSxtr@p@acrshortpl@:new	38
\GLSaccesslong:new	72, 76	\@GLSxtr@p@long@:new	37
\GLSaccesslongpl:new	73, 76	\@GLSxtr@p@longpl@:new	37
\Glsaccesslongpl:new	73	\@GLSxtr@p@plural@:new	36
\glsaccesslongpl:new	72	\@GLSxtr@p@short@:new	36
\GLSaccessname:new	63, 73	\@GLSxtr@p@shortpl@:new	37
\GLSaccessplural:new	65, 74	\@GLSxtr@p@text@:new	36
\GLSaccessshort:new	71, 75	\@GlsXtrEnableOnTheFly:new	15
\GLSaccessshortpl:new	72, 75	\@Glsxtr:new	15
\GLSaccesssymbol:new	68, 74	\@Glsxtr@p@acrlong@:new	38
\GLSaccesssymbolplural:new	69, 74	\@Glsxtr@p@acrlongpl@:new	38
\GLSaccesstext:new	64, 73	\@Glsxtr@p@acrshort@:new	37
\glsentryfmt:moved \glssetabbrvfmt		\@Glsxtr@p@acrshortpl@:new	37
from \glsxtrabbrvfmt to here	19	\@Glsxtr@p@long@:new	37
\GlsXtrEnableInitialTagging:new	90	\@Glsxtr@p@longpl@:new	37
\glsxtrfieldtitlecase:new	81	\@Glsxtr@p@plural@:new	36
\GlsXtrFormatLocationList:new	18	\@Glsxtr@p@short@:new	36
\glsxtrnewabbrevpresetkeyhook:		\@Glsxtr@p@shortpl@:new	36
new	98	\@Glsxtr@p@text@:new	35
\glsxtrtagfont:new	92	\@Glsxtrpl:new	16
\KV@printgloss@nonumberlist:added	18	\@alt@gls@hyp@opt:new	34
\mfu@checkword@do:added	91	\@gls@alt@hyp@opt:new	33
\setabbreviationstyle:added check		\@gls@alt@hyp@opt@char:new	34
for post-definition style switch	112	\@gls@alt@hyp@opt@keys:new	34
0.5.3 (2015-12-09)		\@gls@increment@currunitcount:	
\@glsxtr@autoindex@at:new	87	new	46
\@glsxtr@autoindex@encap:new	88	\@gls@local@increment@currunitcount:	
\@glsxtr@autoindex@esc:new	88	new	47
\@glsxtr@autoindex@level:new	88	\@gls@setdefault@glslink@opts:	
\@glsxtr@autoindex@setname:new	86	new	32
\@glsxtr@doabbreviationsdef:new	6	\@glsxtr:new	15
General: removed \GlsXtrNoGlsWarningNoAutoMakeIndex		\@glsxtr@addunitcounter:new	46
.....	60	\@glsxtr@currunitcount:new	47
\glsdescwidth:added	17	\@glsxtr@ifunitcounter:new	46
\glspagelistwidth:added	18	\@glsxtr@p@acrlong@:new	38
\glsxtrdoautoindexname:new	86	\@glsxtr@p@acrlongpl@:new	38
\glsxtrpostnamehook:new	85	\@glsxtr@p@acrshort@:new	37
\if@glsxtr@format@override:new	85	\@glsxtr@p@acrshortpl@:new	37
\ProvidesGlossariesExtraLang:new	143	\@glsxtr@p@long@:new	37
\RequireGlossariesExtraLang:new	143	\@glsxtr@p@longpl@:new	37

\@glxtr@p@plural@: new	36	\GlsXtrEnableEntryUnitCounting:	
\@glxtr@p@short@: new	36	new	51
\@glxtr@p@shortpl@: new	36	\GlsXtrEnableOnTheFly: new	14
\@glxtr@p@text@: new	35	\Glsxtrpl: new	16
\@glxtr@prevunitcount: new	47	\glxtrpl: new	16
\@glxtr@unitcountlist: new	46	\glxtrpostlocalreset: new	39
\@glxtrpl: new	16	\glxtrpostlocalunset: new	39
\@newglossaryentryposthook: added		\glxtrpostreset: new	39
empty see value if not set and added		\glxtrpostunset: new	38
'see' to field key map	12	\glxtrprotectlinks: new	35
\@sGlsXtrEnableOnTheFly: new	14	\GlsXtrSetAltModifier: new	34
\cGlsformat: added	45	\GlsXtrSetDefaultGlsOpts: new	32
\cglformat: added	45	\glxtrstarflywarn: new	14
\cGlsplformat: added	45	\GlsXtrWarning: new	16
\cglplformat: added	45	\MakeAcronymsAbbreviations: now	
\gl:disablehyper: added	34	disables \setacronymstyle	53
\glsdohyperlink: added	34	0.5.5 (??)	
\glndonohyperlink: added	34	\@glxtr@idx@displaynumberlist:	
\glsenableentryunitcount: new	47	new	57
\glshasattribute: added check for en-		\@glxtr@idx@entrynumberlist: new	58
try's existence	77	\@glxtr@noidx@displaynumberlist:	
\gl:sifattribute: added check for en-		new	57
try's existence	78	\@glxtr@noidx@entrynumberlist:	
\gl:spostlinkhook: added existence		new	58
check	93	\@glxtr@noidx@numberlistloop:	
\Glsxtr: new	15	new	57
\glxtr: new	15	\@glxtr@reg@glosslist: new	54
\glxtrcat: new	15	\makeglossaries: new	54
\glxtrdowrglossaryhook: new	33	1.0 (2016-01-24)	
		\@glxtr@autoindexcrossrefs: new ..	6

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	
\.	93
\@@cGLS@	41, 49
\@@cGLSpl@	41, 49
\@@cGLspl@	41, 49
\@@cgls@	41, 49
\@@cglspl@	41, 42, 49
\@@glo@assign@sortkey	56
\@@glo@no@assign@sortkey	56
\@@glslocalreset	39
\@@glslocalunset	38
\@@glsreset	39
\@@glsunset	38
\@@glsxtr@autoindex@escspch	88, 89
\@@glsxtr@checkspch	86, 87, 89
\@@glsxtr@disabledflycommand	17
\@@newglossaryentry@defcounters	40
\@@newglossaryentry@defunitcounters	47
\@ACRlong	35
\@ACRlongpl	35
\@ACRshort	35
\@ACRshortpl	35
\@Acrlong	35
\@Acrlongpl	35
\@Acrshort	35
\@Acrshortpl	35
\@GLS@	35, 43, 44
\@GLSdesc@	22
\@GLSpl@	35, 43, 44
\@GLSplural@	36
\@GLSsymbol@	23
\@GLStext@	36
\@GLSxtr@full	102
\@GLSxtr@fullpl	103
\@GLSxtr@p@acrlong@	35
\@GLSxtr@p@acrlongpl@	35
\@GLSxtr@p@acrshort@	35
\@GLSxtr@p@acrshortpl@	35
\@GLSxtr@p@long@	35
\@GLSxtr@p@longpl@	35
\@GLSxtr@p@plural@	35
\@GLSxtr@p@short@	35
\@GLSxtr@p@shortpl@	35
\@GLSxtr@p@text@	35
\@GLSxtrlong	35, 106
\@GLSxtrlongpl	35, 108, 109
\@GLSxtrpl	16, 17
\@GLSxtrshort	35, 104
\@GLSxtrshortpl	35, 107
\@GLSxtr@p@acrshortpl@	35
\@GLSxtr@p@long@	35
\@GLSxtr@p@longpl@	35
\@GLSxtr@p@plural@	35
\@GLSxtr@p@short@	35
\@GLSxtr@p@shortpl@	35
\@GLSxtr@p@text@	35
\@GLSxtrlong	35, 106
\@GLSxtrlongpl	35, 108, 109
\@GLSxtrpl	16, 17
\@GLSxtrshort	35, 104
\@GLSxtrshortpl	35, 107

<code>\@acrlong</code>	35	<code>\@gls@entry@count</code>	41
<code>\@acrlongpl</code>	35	<code>\@gls@entry@field</code>	29, 40
<code>\@acrshort</code>	35	<code>\@gls@entry@unitcount</code>	49, 50
<code>\@acrshortpl</code>	35	<code>\@gls@field@link</code>	20–25, 30, 31, 63–71
<code>\@alt@gls@hyp@opt</code>	33	<code>\@gls@hyp@opt</code>	30, 31, 34, 44, 100–109
<code>\@auxout</code>	41, 50, 54, 58, 59	<code>\@gls@hyp@opt@cs</code>	33, 34
<code>\@cGLS</code>	44	<code>\@gls@increment@currcount</code>	40
<code>\@cGLS@</code>	41, 44, 49	<code>\@gls@increment@currunitcount</code>	48
<code>\@cGLSpl</code>	44	<code>\@gls@keymap</code>	12, 29
<code>\@cGLSpl@</code>	41, 44, 49	<code>\@gls@label</code>	33, 112
<code>\@cGLspl@</code>	41, 49	<code>\@gls@levelchar</code>	87
<code>\@cglS@</code>	41, 44, 49	<code>\@gls@link</code>	19, 25–29, 101–109
<code>\@cglSpl@</code>	41, 49	<code>\@gls@link@checkfirsthyper</code>	53
<code>\@disable@onlypremakeg</code>	55	<code>\@gls@link@nocheckfirsthyper</code>	19, 25–29, 101–109
<code>\@do@auxoutstuff</code>	58, 59	<code>\@gls@local@increment@currcount</code>	40
<code>\@do@newglossaryentry</code>	52, 98	<code>\@gls@local@increment@currunitcount</code>	48
<code>\@empty</code>	20, 25–29, 87, 101–109	<code>\@gls@loclist</code>	57, 58
<code>\@end@glsxtr@addunused</code>	13	<code>\@gls@longpl</code>	96–98
<code>\@endfortrue</code>	112	<code>\@gls@noidx@nosanitizesort</code>	56
<code>\@firstofone</code>	85, 91	<code>\@gls@noidx@sanitizesort</code>	56
<code>\@firstofthree</code>	20, 25–28, 33, 34, 101, 102, 104, 105, 107, 108	<code>\@gls@noidx@loclist@finalsep</code>	57
<code>\@firstoftwo</code>	20–23, 26–29, 31, 33, 94, 95, 101–103, 107–109	<code>\@gls@noidx@loclist@prev</code>	57
<code>\@for</code>	13, 40, 51, 54, 56, 81, 90	<code>\@gls@noidx@loclist@sep</code>	57
<code>\@glo@assign@sortkey</code>	56	<code>\@gls@org@glsnoidx@displayloc</code>	57, 58
<code>\@glo@category</code>	45	<code>\@gls@org@glsseeformat</code>	57, 58
<code>\@glo@countunit</code>	45	<code>\@gls@preglossaryhook</code>	91
<code>\@glo@default@sorttype</code>	56	<code>\@gls@quotechar</code>	86
<code>\@glo@label</code>	12, 29	<code>\@gls@reference</code>	54
<code>\@glo@name</code>	86	<code>\@gls@setdefault@glslink@opts</code>	32
<code>\@glo@see</code>	12, 13	<code>\@gls@short</code>	97
<code>\@glo@sort</code>	86	<code>\@gls@shortpl</code>	96–98
<code>\@glo@sorttype</code>	56	<code>\@gls@tmpb</code>	89
<code>\@glo@tmp</code>	29	<code>\@gls@type</code>	56, 112
<code>\@glo@type</code>	12, 52, 54, 56, 58, 59, 61, 62	<code>\@gls@write@entrycounts</code>	41
<code>\@glo@types</code>	79	<code>\@gls@write@entryunitcounts</code>	49
<code>\@gls@</code>	35, 42, 44	<code>\@gls@write@entryunitcounts@do</code>	50
<code>\@gls@actualchar</code>	87	<code>\@glsabbrv@current@abbreviation</code>	97, 109
<code>\@gls@alt@hyp@opt</code>	34	<code>\@glsacronymlists</code>	52
<code>\@gls@alt@hyp@opt@char</code>	33, 34	<code>\@glsentry</code>	41, 50
<code>\@gls@alt@hyp@opt@keys</code>	34	<code>\@glsfirstplural@</code>	21
<code>\@gls@automake</code>	56	<code>\@glslink</code>	34, 35
<code>\@gls@checkedmkidx</code>	86, 87, 89	<code>\@glsnumberformat</code>	85, 86
<code>\@gls@checkmkidxchars</code>	86	<code>\@glsorder</code>	54
<code>\@gls@codepage</code>	59	<code>\@glspl@</code>	35, 42, 44
<code>\@gls@declareoption</code>	4	<code>\@glsplural@</code>	36, 65
<code>\@gls@doautomake</code>	56	<code>\@gls@punc@token</code>	95
<code>\@gls@encapchar</code>	87	<code>\@gls@target</code>	34
		<code>\@gls@text@</code>	35

<code>\@glsxtr</code>	15, 17	<code>\@glsxtr@gobbleto@endescspch</code>	89
<code>\@glsxtr@abbreviationsdef</code>	6, 9, 10	<code>\@glsxtr@idx@displaynumberlist</code>	55
<code>\@glsxtr@activate@initialtagging</code>	91, 92	<code>\@glsxtr@idx@entrynumberlist</code>	55
<code>\@glsxtr@addunitcounter</code>	45	<code>\@glsxtr@ifcsstart</code>	14
<code>\@glsxtr@addunusedxrefs</code>	12, 13	<code>\@glsxtr@ifpunctoken</code>	95
<code>\@glsxtr@attrval</code>	86	<code>\@glsxtr@ifunitcounter</code>	45
<code>\@glsxtr@autoindex@at</code>	86–88	<code>\@glsxtr@insert@dots</code>	96
<code>\@glsxtr@autoindex@doextra@esc</code>	86	<code>\@glsxtr@insert@dots@next</code>	97
<code>\@glsxtr@autoindex@encap</code>	86–88	<code>\@glsxtr@insertdots</code>	97
<code>\@glsxtr@autoindex@esc</code>	86–89	<code>\@glsxtr@label</code>	13, 81
<code>\@glsxtr@autoindex@escat</code>	87, 88	<code>\@glsxtr@noidx@displaynumberlist</code> ...	55
<code>\@glsxtr@autoindex@escencap</code>	87, 88	<code>\@glsxtr@noidx@entrynumberlist</code>	55
<code>\@glsxtr@autoindex@esclevel</code>	87, 88	<code>\@glsxtr@noidx@numberlistloop</code>	55
<code>\@glsxtr@autoindex@escquote</code>	86, 88	<code>\@glsxtr@notfoundinlist</code>	95
<code>\@glsxtr@autoindex@level</code>	87, 88	<code>\@glsxtr@optlist</code>	16
<code>\@glsxtr@autoindex@setname</code>	86	<code>\@glsxtr@org@Glsxtrtitlefirst</code>	134
<code>\@glsxtr@autoindex@crossrefs</code>	5, 12	<code>\@glsxtr@org@Glsxtrtitlefirstplural</code>	134
<code>\@glsxtr@cat</code>	40, 51, 90	<code>\@glsxtr@org@Glsxtrtitleplural</code>	134
<code>\@glsxtr@csname</code>	46, 47, 49	<code>\@glsxtr@org@Glsxtrtitleshort</code>	134
<code>\@glsxtr@currentunitcount</code>	46, 47, 49	<code>\@glsxtr@org@Glsxtrtitleshortpl</code> ...	134
<code>\@glsxtr@currunitcount</code>	48, 50	<code>\@glsxtr@org@Glsxtrtitletext</code>	134
<code>\@glsxtr@declareoption</code>	4, 6, 9	<code>\@glsxtr@org@MakeUppercase</code>	134
<code>\@glsxtr@defaultnoglossarywarning</code> ...	9	<code>\@glsxtr@org@checkfirsthyper</code>	31, 53
<code>\@glsxtr@disabledflycommand</code>	17	<code>\@glsxtr@org@Glsxtrtitlefirst</code>	134
<code>\@glsxtr@do@@windex</code>	33	<code>\@glsxtr@org@Glsxtrtitlefirstplural</code>	134
<code>\@glsxtr@do@glsglisablehyperinlist</code>	31, 32	<code>\@glsxtr@org@Glsxtrtitleplural</code>	134
<code>\@glsxtr@do@titlecaps@warn</code> ...	82–84, 91	<code>\@glsxtr@org@Glsxtrtitleshort</code>	134
<code>\@glsxtr@doabbreviationsdef</code>	6	<code>\@glsxtr@org@Glsxtrtitleshortpl</code> ...	134
<code>\@glsxtr@doaccsupp</code>	9	<code>\@glsxtr@org@Glsxtrtitletext</code>	134
<code>\@glsxtr@dostylewarn</code>	112	<code>\@glsxtr@org@makeglossaries</code>	54
<code>\@glsxtr@enabletagging</code>	90	<code>\@glsxtr@org@markboth</code>	133
<code>\@glsxtr@end@</code>	14	<code>\@glsxtr@org@markright</code>	133
<code>\@glsxtr@endescspch</code>	87–89	<code>\@glsxtr@org@newacronymstyle</code>	53
<code>\@glsxtr@entrycount@org@localreset</code> ..	41	<code>\@glsxtr@org@postdescription</code>	92
<code>\@glsxtr@entrycount@org@localunset</code> ..	40	<code>\@glsxtr@org@setacronymstyle</code>	53
<code>\@glsxtr@entrycount@org@reset</code> ...	40, 41	<code>\@glsxtr@org@printglossary</code>	16, 17
<code>\@glsxtr@entrycount@org@unset</code>	40	<code>\@glsxtr@p@acrlong@</code>	35
<code>\@glsxtr@entryunitcount@org@localreset</code>	49	<code>\@glsxtr@p@acrlongpl@</code>	35
<code>\@glsxtr@entryunitcount@org@localunset</code>	48	<code>\@glsxtr@p@acrshort@</code>	35
<code>\@glsxtr@entryunitcount@org@reset</code> ..	48	<code>\@glsxtr@p@acrshortpl@</code>	35
<code>\@glsxtr@entryunitcount@org@unset</code> ..	48	<code>\@glsxtr@p@long@</code>	35
<code>\@glsxtr@field@linkdefs</code>	19	<code>\@glsxtr@p@longpl@</code>	35
<code>\@glsxtr@format@overridefalse</code>	85	<code>\@glsxtr@p@plural@</code>	35
<code>\@glsxtr@format@overridetrue</code>	85, 86	<code>\@glsxtr@p@short@</code>	35
<code>\@glsxtr@foundinlist</code>	95	<code>\@glsxtr@p@shortpl@</code>	35
<code>\@glsxtr@full</code>	100	<code>\@glsxtr@p@text@</code>	35
<code>\@glsxtr@fullpl</code>	102	<code>\@glsxtr@prevunitcount</code>	48
		<code>\@glsxtr@reg@glosslist</code>	54–56
		<code>\@glsxtr@setentrycountunsetattr</code>	39

\@glxtr@setentryunitcountunsetattr	51	A	
\@glxtr@setupshortcuts	8–10	\AB	7
\@glxtr@swaptwo	95	\Ab	7
\@glxtr@tag	91	\ab	7
\@glxtr@taggingcs	91	abbreviation styles:	
\@glxtr@type	81	short	121
\@glxtr@unitcountlist	46	\abbreviationsname	6
\@glxtrdocdeffalse	13	\abbrvpluralsuffix	
\@glxtrindexcrossrefsfalse	6	..	97, 98, 114, 116, 117, 119–122, 124–130
\@glxtrindexcrossrefstrue	6	\ABP	7
\@glxtrlong	35, 105	\Abp	7
\@glxtrlongpl	35, 108	\abp	7
\@glxtrpl	16, 17	\ACRfullfmt	52
\@glxtrshort	35, 104	\Acrfullfmt	52
\@glxtrshortpl	35, 106, 107	\acrfullfmt	52
\@glxtrundeftag	5, 10	\ACRfullplfmt	52
\@gobble	6, 97	\Acrfullplfmt	52
\@ifnextchar	33	\acrfullplfmt	52
\@ifpackageloaded	4, 6, 63, 81, 82, 84, 85, 143	\acronymentry	52
\@ifstar	14, 33, 90	\acronymfont	25–29, 37, 38, 53
\@ifundefined	143	\acronymname	6
\@input@	58	\acronymsort	52
\@istfilename	54	\acronymtype	6, 52, 53
\@makeglossary	54, 55	\acrpluralsuffix	52
\@mfu@domakefirsttuc	91	\actualchar	89
\@mfu@nocaplist	91	\advance	41, 50
\@one	41, 50	\AF	7
\@newglossaryentry@defcounters	.. 40, 47	\Af	7
\@newglossaryentryposthook	29	\af	7
\@newglossaryentryprehook	29	\AFP	7
\@nnil	87, 89, 95, 96	\Afp	7
\@no@makeglossaries	62	\afp	7
\@onelevel@sanitize	16	\AL	7
\@onlypreamble	17, 50, 86, 88, 90	\Al	7
\@printgloss@setsort	56	\al	7
\@printglossary	16, 17	\ALP	7
\@sGlsXtrEnableOnTheFly	14	\Alp	7
\@secondofthree	20,	\alp	7
21, 25–28, 30, 101, 103, 104, 106, 107, 109		\AnyTrackedLanguages	144
\@secondoftwo	.. 20, 22–29, 31, 34, 95, 101, 102, 104–109	\appto	12, 29, 33, 40, 47, 85, 94, 97
\@thirdofthree	20,	\AS	7
21, 26–29, 31, 102, 103, 105, 106, 108, 109		\As	7
\@thirdoftwo	22–25	\as	7
\@warn@nomakeglossaries	59	\ASP	7
\@xdy@main@language	58	\Asp	7
\@xdy@language	58, 59	\asp	7
_	60, 61	\AtBeginDocument	10, 17, 18
		\AtEndDocument	12, 41, 49, 58, 59

B		
babel package	86, 87, 94	
\begin	44, 60, 61	
C		
category attributes:		
discardperiod	94	
entrycount	38–41, 51	
firsttuc	84	
glossdesc	81	
glossname	82	
headuc	135	
indexname	86	
indexonlyfirst	32	
insertdots	97	
nohyper	31	
nohyperfirst	117	
regular	19, 45, 114–117, 119, 121–123	
\cGLS	7, 39, 51	
\cGls	7, 39, 51	
\cglS	7, 39, 51	
\cGLSformat	43	
\cGlsformat	43	
\cglSformat	42, 44	
\cGLSpl	7, 39, 51	
\cGlspl	7, 39, 51	
\cglSpl	7, 39, 51	
\cGLSplformat	43	
\cGlsplformat	43	
\cglSplformat	42, 45	
\columnwidth	17, 18	
\count@	41, 42, 50	
\cs	44	
\csdef	29–31, 41, 46, 47, 49, 76, 112, 113, 119	
\csedef	47	
\csgdef	41, 46, 49	
\csname	19, 25–31, 47, 58, 59, 61, 62, 81, 96, 101–109, 113	
\csuse	30, 45–49, 77, 92, 93, 112	
\csxdef	12, 46, 49	
\CurrentOption	9	
\CurrentTrackedTag	144	
\CustomAbbreviationFields	98, 114–118, 120–123	
D		
\DeclareAcronymList	52	
\DeclareOption	4	
\DeclareOptionX	4, 9	
\def	10, 13–29, 34–38, 42–44, 52, 55–57, 63–71, 85–89, 91, 94–98, 101–109, 112	
\define@boolkey	5, 32	
\define@choicekey	5, 8, 9	
\define@key	29, 96	
\DefineAcronymSynonyms	8	
\detokenize	14	
\dimen@	54	
\dimexpr	17, 18	
\disable@keys	6, 10, 13	
\do	13, 40, 51, 54, 56, 81, 90	
\do@glS@link@checkfirsthyper	19, 25–29, 101–109	
\do@glSdisablehyperinlist	32	
doc package	89	
\DTLifinlist	55, 56	
E		
\eappto	86	
\edef	31, 45–47, 49, 54, 56, 58, 59, 86, 87, 89, 96	
\else	5, 6, 8, 9, 14, 18, 19, 32, 42, 54, 56, 60, 62, 85–87, 89, 95, 97	
\emph	130–132	
\encapchar	89	
\end	60, 61	
\endcsname	19, 25–31, 47, 58, 59, 61, 62, 81, 96, 101–109, 113	
entry categories:		
abbreviation	109	
general	76, 78	
index	79	
\epreto	86	
\equal	62	
etoolbox package	4	
\expandafter	9, 13–16, 29–31, 33, 34, 44, 45, 55, 56, 81, 83, 84, 86, 89, 95, 97, 98	
\expandonce	52, 86, 87	
F		
\fi	5, 6, 8, 9, 12, 14, 17–19, 32, 33, 42, 49, 50, 54, 56, 59, 60, 62, 86, 87, 89, 95, 97	
\firstacronymfont	53, 54	
\footnote	117–119	
\forallglossaries	12, 79, 81	
\forallglSentries	41, 50	
\ForEachTrackedDialect	144	
\forallglSentries	12, 79, 81	
\foralllistcsloop	50	
\foralllistloop	57, 91	
\futurelet	95	

G

<code>\gdef</code>	88	<code>\glsaccesslong</code>	27, 28, 99, 105, 106, 114, 116, 118–123
<code>\Genacrfullformat</code>	52	<code>\Glsaccesslongpl</code> ..	28, 99, 109, 115, 120, 123	
<code>\genacrfullformat</code>	52	<code>\glsaccesslongpl</code>	28,	
<code>\GenericAcronymFields</code>	52		29, 99, 108, 109, 114, 116, 118–120, 122, 123	
<code>\Genplacrfullformat</code>	52, 53	<code>\GLSaccessname</code>	64	
<code>\genplacrfullformat</code>	52	<code>\Glsaccessname</code>	63	
<code>\glo@name</code>	83, 84	<code>\glsaccessname</code>	63	
<code>glossaries package</code>	4, 5, 7–11, 19,	<code>\GLSaccessplural</code>	65	
	31, 32, 34, 38, 44, 51, 53, 54, 84, 92, 96, 133	<code>\Glsaccessplural</code>	65	
<code>glossaries-accsupp package</code>	9, 63	<code>\glsaccessplural</code>	65	
<code>glossaries-extra package</code>	2, 63	<code>\Glsaccessshort</code>		
<code>\GlossariesExtraWarning</code>			25, 104, 110, 116, 118–120, 122	
	5, 6, 14, 16, 53, 60, 91	<code>\glsaccessshort</code>		
<code>\GlossariesExtraWarningNoLine</code> ..	6, 42, 50		25, 26, 99, 104, 105, 110, 114–116, 118–123	
<code>\GlossariesWarning</code>	57, 58, 112	<code>\Glsaccessshortpl</code>		
<code>\GlossariesWarningNoLine</code>	55, 59		26, 107, 110, 116, 118–120, 122	
<code>\glossaryentrynumbers</code>	19	<code>\glsaccessshortpl</code>		
<code>\glossarysection</code>	61		26, 27, 99, 107, 108, 110, 114–116, 118–123	
<code>\glossarytitle</code>	61	<code>\GLSaccesssymbol</code>	68	
<code>\glossarytoctitle</code>	61	<code>\Glsaccesssymbol</code>	68, 90	
<code>\GLS</code>	39, 51	<code>\glsaccesssymbol</code>	67, 90, 94	
<code>\Gls</code>	15, 39, 51	<code>\GLSaccesssymbolplural</code>	69	
<code>\gls</code>	15, 17, 39, 51, 60	<code>\Glsaccesssymbolplural</code>	69	
<code>\gls@assign@field</code>	29	<code>\glsaccesssymbolplural</code>	68	
<code>\gls@checkseeallowed</code>	13, 55	<code>\GLSacesstext</code>	64	
<code>\gls@codepage</code>	59	<code>\Glsacesstext</code>	64	
<code>\gls@defdocnewglossaryentry</code>	40, 48	<code>\glsacesstext</code>	64	
<code>\gls@defglossaryentry</code>	15, 16	<code>\glsacrshortcutstrue</code>	8	
<code>\gls@save@numberlist</code>	18, 19	<code>\glsacspacemax</code>	54	
<code>\glsabbrvdefaultfont</code>		<code>\glsadd</code>	13, 60	
	100, 114, 116, 117, 119–122	<code>\glsaddstoragekey</code>	76	
<code>\glsabbrvfont</code>	36, 37,	<code>\glsbackslash</code>	14	
	53, 100, 104, 105, 107, 108, 110, 114–132	<code>\glscapscase</code>	20–31, 101–111	
<code>\glsabrvfont</code>	114–117, 119	<code>\glscategory</code>	19, 31, 36, 37,	
<code>\GLSaccessdesc</code>	70		77–79, 81–85, 90, 92, 93, 101–105, 107, 108	
<code>\Glsaccessdesc</code>	70, 82, 90	<code>\glscategorylabel</code>	31, 96–98, 119	
<code>\glsaccessdesc</code>	69, 82, 93	<code>\glsclosebrace</code>	61, 62	
<code>\GLSaccessdescplural</code>	71	<code>\glscurrententrylabel</code>	92	
<code>\Glsaccessdescplural</code>	70	<code>\glscustomtext</code>	19, 25–29, 101–109, 111	
<code>\glsaccessdescplural</code>	70	<code>\glsdefaulttype</code>	6, 60	
<code>\GLSaccessfirst</code>	66	<code>\glsdescriptionaccessdisplay</code> ..	69, 70, 82	
<code>\Glsaccessfirst</code>	66	<code>\glsdescriptionpluralaccessdisplay</code>		
<code>\glsaccessfirst</code>	66		70, 71	
<code>\GLSaccessfirstplural</code>	67	<code>\glsdetoklabel</code>	11,	
<code>\Glsaccessfirstplural</code>	67		13, 14, 40, 41, 46–50, 57, 58, 81, 83, 84	
<code>\glsaccessfirstplural</code>	67	<code>\glsdisplaynumberlist</code>	55, 57	
<code>\Glsaccesslong</code>	27, 99, 106, 115, 120, 123	<code>\glsdohyperlink</code>	35	
		<code>\glsdoifexists</code> ...	19, 25–29, 57, 58, 101–109	

<code>\glsdoifexistsorwarn</code>	81, 82, 84, 85, 89, 90	<code>\glsentryuserv</code>	24
<code>\glsdonohyperlink</code>	34, 35	<code>\Glsentryuservi</code>	25
<code>\glsdosanitizesort</code>	56	<code>\glsentryuservi</code>	25
<code>\glsenableentrycount</code>	39, 42, 50	<code>\glsextrinlinefullformat</code>	121
<code>\glsenableentryunitcount</code>	41, 50, 51	<code>\glsfieldxdef</code>	81
<code>\glsentrycurrcount</code>	40, 41, 48	<code>\GLSfirst</code>	138
<code>\Glsentrydesc</code>	22, 69, 75, 82	<code>\Glsfirst</code>	139
<code>\glsentrydesc</code>	22, 69, 70, 75, 82	<code>\glsfirst</code>	138
<code>\Glsentrydescplural</code>	22, 70, 75	<code>\glsfirstabbrvdefaultfont</code>	100, 114, 116, 118–122
<code>\glsentrydescplural</code>	22, 70, 71, 75	<code>\glsfirstabbrvfont</code>	53, 99, 114–123
<code>\Glsentryfirst</code>	20, 45, 66, 74	<code>\glsfirstaccessdisplay</code>	65, 66
<code>\glsentryfirst</code>	20, 45, 65, 66, 74, 142	<code>\glsfirstlongdefaultfont</code>	100, 114, 116, 118–121, 123
<code>\Glsentryfirstplural</code>	21, 45, 67, 74	<code>\glsfirstlongfont</code>	99, 114–123
<code>\glsentryfirstplural</code>	21, 45, 66, 67, 74, 143	<code>\GLSfirstplural</code>	139
<code>\Glsentryfull</code>	52	<code>\Glsfirstplural</code>	139
<code>\glsentryfull</code>	52	<code>\glsfirstplural</code>	139
<code>\Glsentryfullpl</code>	53	<code>\glsfirstpluralaccessdisplay</code>	66, 67
<code>\glsentryfullpl</code>	52	<code>\glsforeachincategory</code>	112
<code>\Glsentrylong</code>	37, 38, 45, 72, 76	<code>\glsгенentryfmt</code>	19
<code>\glsentrylong</code>	37, 38, 45, 72, 75, 76, 119	<code>\glsgetattribute</code>	42, 46–48, 86
<code>\Glsentrylongpl</code>	37, 38, 45, 73, 76	<code>\glsgetcategoryattribute</code>	77
<code>\glsentrylongpl</code>	37, 38, 45, 72, 73, 76	<code>\glshasattribute</code>	41, 42, 46, 47, 49, 50, 86, 114–117, 119
<code>\Glsentryname</code>	22, 63, 73, 83–85	<code>\glshascategoryattribute</code>	77
<code>\glsentryname</code>	22, 63, 73, 86	<code>\glshypernumber</code>	85
<code>\glsentrynumberlist</code>	55, 58	<code>\glsifattribute</code>	32, 79, 81–84, 92, 94, 135–139
<code>\Glsentryplural</code>	21, 65, 73	<code>\glsifcategory</code>	79
<code>\glsentryplural</code>	20, 21, 65, 73, 74, 141, 142	<code>\glsifcategoryattribute</code>	31, 78, 97, 98
<code>\glsentryprevcount</code>	40, 42, 48	<code>\glsifplural</code>	20–23, 25–29, 94, 101–110
<code>\glsentryprevmaxcount</code>	48	<code>\glsifregular</code>	19, 45
<code>\glsentryprevtotalcount</code>	48	<code>\glsifregularcategory</code>	79
<code>\Glsentryshort</code>	36, 37, 71, 75	<code>\glsinsert</code>	20, 25–29, 101–111
<code>\glsentryshort</code>	36, 37, 54, 71, 75, 140	<code>\glskeylisttok</code>	52, 97, 98
<code>\Glsentryshortpl</code>	37, 72, 75	<code>\glslabel</code>	19, 31, 32, 93, 94, 110, 111, 119
<code>\glsentryshortpl</code>	36–38, 71, 72, 75, 140, 141	<code>\glslabeltok</code>	52, 97, 98, 114–117, 119–122, 124
<code>\Glsentrysymbol</code>	23, 68, 74	<code>\glsletentryfield</code>	86
<code>\glsentrysymbol</code>	23, 67, 68, 74	<code>\glslink</code>	52
<code>\Glsentrysymbolplural</code>	23, 69, 74	<code>\glslink options</code>	format 85, noindex 32
<code>\glsentrysymbolplural</code>	23, 68, 69, 74	<code>\glslinkcheckfirsthyperhook</code>	31
<code>\Glsentrytext</code>	20, 64, 73	<code>\glslinkvar</code>	33, 34
<code>\glsentrytext</code>	20, 64, 73, 141	<code>\glslongaccessdisplay</code>	72
<code>\Glsentryuseri</code>	23	<code>\glslongpltok</code>	98, 114–117, 119, 122, 123
<code>\glsentryuseri</code>	23	<code>\glslongpluralaccessdisplay</code>	72, 73
<code>\Glsentryuserii</code>	24	<code>\glslongtok</code>	52, 97, 98, 114–123
<code>\glsentryuserii</code>	24	<code>\glsnameaccessdisplay</code>	63, 83, 84
<code>\Glsentryuseriii</code>	24		
<code>\glsentryuseriii</code>	24		
<code>\Glsentryuseriv</code>	24		
<code>\glsentryuseriv</code>	24		
<code>\Glsentryuserv</code>	24		

<code>\glsnamefont</code>	83–85	<code>\glsxtr@doooption</code>	4, 6, 9
<code>\glsnoidxdisplayloc</code>	57, 58	<code>\glsxtr@ifnextpunc</code>	95
<code>\glsnoidxdisplayloclisthandler</code>	57	<code>\glsxtr@ifpunctoken</code>	95
<code>\glsnoidxloclist</code>	58	<code>\glsxtr@keylist</code>	15, 16
<code>\glsnoidxnumberlistloophandler</code>	57	<code>\glsxtr@next</code>	95
<code>\glsnonumberlistfalse</code>	18	<code>\glsxtr@orgmakenoidxglossaries</code>	13
<code>\glsnonumberlisttrue</code>	18	<code>\glsxtr@punclist</code>	94, 95
<code>\glsnumberlistloop</code>	55	<code>\glsxtr@warnonexistsordo</code>	5, 11, 12
<code>\glsnumlistlastsep</code>	57	<code>\glsxtrabbrvtype</code>	6, 98
<code>\glsnumlistsep</code>	57	<code>\glsxtraddallcrossrefs</code>	12
<code>\glsopenbrace</code>	61, 62	<code>\glsxtrcat</code>	15, 16
<code>\glsorder</code>	54	<code>\GlsXtrDefineAbbreviationShortcuts</code> ..	8
<code>\GLSpl</code>	39, 51	<code>\GlsXtrDefineOtherShortcuts</code>	8
<code>\Glspl</code>	16, 39, 51	<code>\glsxtrdiscardperiod</code>	93
<code>\glspl</code>	16, 39, 51	<code>\glsxtrdoautoindexname</code>	32, 33, 85
<code>\GLSplural</code>	137, 138	<code>\glsxtrdopostpunc</code>	119
<code>\Glsplural</code>	138	<code>\glsxtrdowrglossaryhook</code>	33
<code>\glsplural</code>	137	<code>\GlsXtrEnableEntryCounting</code>	51
<code>\glspluralaccessdisplay</code>	65	<code>\GlsXtrEnableEntryUnitCounting</code>	39
<code>\glspluralsuffix</code>		<code>\GlsXtrEnableOnTheFly</code>	14, 17
. 97, 100, 114, 116, 117, 119–122, 124, 127		<code>\glsxtrfieldtitlecase</code>	82–84
<code>\glspostdescription</code>	92	<code>\GlsXtrFormatLocationList</code>	18, 19
<code>\glspostlinkhook</code>	19, 25–29, 101–109	<code>\GLSxtrfull</code>	7
<code>\glsprestandardsort</code>	55	<code>\Glsxtrfull</code>	7
<code>\glseeformat</code>	57, 58	<code>\glsxtrfull</code>	7
<code>\glssetabbrvfmt</code>		<code>\Glsxtrfullformat</code>	
. 19, 36, 37, 81–85, 90, 101–105, 107, 108		. 99, 111, 113, 115, 116, 118, 119, 121–123	
<code>\glssetattribute</code> ... 114–117, 119–122, 124		<code>\glsxtrfullformat</code>	
<code>\glssetcategoryattribute</code> 99, 111, 113–116, 118, 119, 121–123	
. 40, 51, 53, 77, 78, 80, 90		<code>\GLSxtrfullpl</code>	7
<code>\glsshortaccessdisplay</code>	71	<code>\Glsxtrfullpl</code>	7
<code>\glsshortpltok</code>	98, 114–117, 119–121	<code>\glsxtrfullpl</code>	7
<code>\glsshortpluralaccessdisplay</code>	71, 72	<code>\Glsxtrfullplformat</code>	
<code>\glsshorttok</code> 52, 97, 98, 114–118, 120, 121, 123		. 99, 111, 113, 115, 116, 118, 119, 121–123	
<code>\glssymbolaccessdisplay</code>	67, 68	<code>\glsxtrfullplformat</code>	
<code>\glssymbolpluralaccessdisplay</code> ...	68, 69	. 111, 113, 114, 116, 118, 119, 121–123	
<code>\GLStext</code>	136, 137	<code>\glsxtrfullsep</code>	99, 114–123
<code>\Glstext</code>	137	<code>\glsxtrgenabbrvfmt</code>	19
<code>\glstext</code>	137	<code>\Glsxtrheadfirst</code>	134
<code>\glstextaccessdisplay</code>	64	<code>\glsxtrheadfirst</code>	134
<code>\glstextup</code>	124	<code>\Glsxtrheadfirstplural</code>	134
<code>\glstype</code>	25–29, 101–109	<code>\glsxtrheadfirstplural</code>	134
<code>\glsunset</code>	13, 42, 43	<code>\Glsxtrheadplural</code>	134
<code>\glswrite</code>	54	<code>\glsxtrheadplural</code>	134
<code>\Glsxtr</code>	17	<code>\Glsxtrheadshort</code>	134
<code>\glsxtr</code>	17	<code>\glsxtrheadshort</code>	134
<code>\glsxtr@addunused</code>	13	<code>\Glsxtrheadshortpl</code>	134
<code>\glsxtr@applyabbrvfmt</code>	109	<code>\glsxtrheadshortpl</code>	134
<code>\glsxtr@applyabbrvstyle</code>	96, 97, 112	<code>\Glsxtrheadtext</code>	134

<code>\glxtrheadtext</code>	134	<code>\glxtrpostunset</code>	38, 40, 48
<code>\glxtrtrifcounttrigger</code>	42, 43	<code>\glxtrprotectlinks</code>	34
<code>\glxtrtrifemptyglossary</code>	61	<code>\glxtrrestoremarkhook</code>	133
<code>\glxtrtrifindexing</code>	32	<code>\glxtrtrscfont</code>	124–127
<code>\glxtrtrifnextpunc</code>	94, 95	<code>\glxtrtrscsuffix</code>	124–127
<code>\glxtrtrifperiod</code>	94	<code>\GlsXtrSetActualChar</code>	89
<code>\glxtrtrifwasfirstuse</code>		<code>\GlsXtrSetEncapChar</code>	89
.. 20, 21, 25–29, 31, 93, 101, 104–109, 119		<code>\GlsXtrSetEscChar</code>	89
<code>\Glsxtrinlinefullformat</code>		<code>\GlsXtrSetLevelChar</code>	89
..... 100, 101, 113, 118–120, 122, 123		<code>\glxtrsetupfulldefs</code>	101–103
<code>\glxtrinlinefullformat</code>		<code>\GLSxtrshort</code>	7, 135, 136
..... 100–102, 113, 118–121, 123		<code>\Glsxtrshort</code>	7, 136
<code>\Glsxtrinlinefullplformat</code>		<code>\glxtrshort</code>	7, 135
..... 100, 103, 113, 118, 120, 122, 123		<code>\GLSxtrshortpl</code>	7, 135, 136
<code>\glxtrinlinefullplformat</code>		<code>\Glsxtrshortpl</code>	7, 136
..... 99, 100, 102, 103, 113, 118–121, 123		<code>\glxtrshortpl</code>	7, 135
<code>\GLSxtrlong</code>	7	<code>\glxtrtrsmfont</code>	127–130
<code>\Glsxtrlong</code>	7	<code>\glxtrtrsmsuffix</code>	127–130
<code>\glxtrlong</code>	7	<code>\glxtrtrtagfont</code>	92
<code>\GLSxtrlongpl</code>	7	<code>\Glsxtrtrtitlefirst</code>	134, 142
<code>\Glsxtrlongpl</code>	7	<code>\glxtrtrtitlefirst</code>	134, 142
<code>\glxtrlongpl</code>	7	<code>\Glsxtrtrtitlefirstplural</code>	134, 143
<code>\glxtrmarkhook</code>	133	<code>\glxtrtrtitlefirstplural</code>	134, 143
<code>\glxtrnewabbrevpresetkeyhook</code>	98	<code>\Glsxtrtrtitleplural</code>	134, 142
<code>\glxtrnewnumber</code>	7	<code>\glxtrtrtitleplural</code>	134, 141, 142
<code>\glxtrnewsymbol</code>	7	<code>\Glsxtrtrtitleshort</code>	134, 140
<code>\glxtrNoGlossaryWarning</code>	9, 58	<code>\glxtrtrtitleshort</code>	134, 140
<code>\GlsXtrNoGlsWarningAutoMake</code>	62	<code>\Glsxtrtrtitleshortpl</code>	134, 141
<code>\GlsXtrNoGlsWarningBuildInfo</code>	62	<code>\glxtrtrtitleshortpl</code>	134, 140
<code>\GlsXtrNoGlsWarningCheckFile</code>	62	<code>\Glsxtrtrtitletext</code>	134, 141
<code>\GlsXtrNoGlsWarningEmptyMain</code>	62	<code>\glxtrtrtitletext</code>	134, 141
<code>\GlsXtrNoGlsWarningEmptyNotMain</code>	62	<code>\glxtrtrundefaction</code>	5, 10–12
<code>\GlsXtrNoGlsWarningEmptyStart</code>	62	<code>\glxtrtrundeftag</code>	10
<code>\GlsXtrNoGlsWarningHead</code>	61	<code>\GlsXtrUseAbbrStyleFmts</code>	115, 117, 124–132
<code>\GlsXtrNoGlsWarningMisMatch</code>	62	<code>\GlsXtrUseAbbrStyleSetup</code>	124–132
<code>\GlsXtrNoGlsWarningNoOut</code>	62	<code>\GlsXtrWarning</code>	15, 16
<code>\GlsXtrNoGlsWarningTail</code>	62		
<code>\Glsxtrpl</code>	17		
<code>\glxtrpl</code>	17		
<code>\glxtrpostdescription</code>	80, 92		
<code>\glxtrpostlink</code>	93		
<code>\glxtrpostlinkendsentence</code>	93		
<code>\glxtrpostlinkhook</code>	93		
<code>\glxtrpostlocalreset</code>	39, 41, 49		
<code>\glxtrpostlocalunset</code>	38, 40, 48		
<code>\glxtrpostnamehook</code>	83–85		
<code>\GlsXtrPostNewAbbreviation</code>			
..... 98, 113–117, 119–122, 124			
<code>\glxtrpostreset</code>	39, 40, 48, 49		

H

<code>\hsize</code>	17, 18
<code>\hyperlink</code>	34
<code>\hyperpage</code>	85
<code>hyperref package</code>	34, 85, 133, 140

I

<code>\if</code>	14
<code>\if@glxtr@format@override</code>	86
<code>\if@glxtr@docdef</code>	13
<code>\if@glxtr@index@crossrefs</code>	5, 12
<code>\ifblank</code>	15, 16, 54
<code>\ifcase</code>	5, 8, 9

<code>\newglossaryentry</code> ...	7, 14, 40, 48, 52, 80, 98	<code>\ns@glstrshortpl</code>	106
<code>\newglossaryentry options</code>		<code>\null</code>	9
<code>desc</code>	69, 70, 75	<code>\number</code>	47–49
<code>descplural</code>	70, 71, 75	<code>\numexpr</code>	47, 49
<code>first</code>	34, 65, 66, 74, 114, 138, 139, 142		
<code>firstplural</code>	66, 67, 74, 114, 139, 142		
<code>hyper</code>	133		
<code>long</code>	72, 76		
<code>longplural</code>	73, 76		
<code>name</code>	63, 73, 86		
<code>noindex</code>	133		
<code>plural</code>	65, 73, 74, 114, 137, 138, 141		
<code>see</code>	6, 13, 55		
<code>short</code>	71, 75, 96		
<code>shortplural</code>	72, 75, 96		
<code>symbol</code>	67, 68, 74		
<code>symbolplural</code>	68, 69, 74		
<code>text</code>	34, 64, 73, 114, 136, 137, 141		
<code>\newif</code>	85		
<code>\newnum</code>	7		
<code>\newrobustcmd</code>			
.....	30, 31, 44, 91, 92, 100–109, 135–139		
<code>\newsym</code>	7		
<code>\newterm</code>	79		
<code>\newtoks</code>	96		
<code>\newwrite</code>	54		
<code>\NoCaseChange</code>	135–139		
<code>\noexpand</code>	52, 58, 59, 86, 87, 98		
<code>\nofiles</code>	61		
<code>\noindent</code>	62		
<code>\nopostdesc</code>	15, 16, 80		
<code>\nr</code>	5, 8, 9		
<code>\ns@GLSxtrfull</code>	102		
<code>\ns@GLSxtrfull</code>	101		
<code>\ns@glstrfull</code>	100		
<code>\ns@GLSxtrfullpl</code>	103		
<code>\ns@GLSxtrfullpl</code>	102		
<code>\ns@glstrfullpl</code>	102		
<code>\ns@GLSxtrlong</code>	106		
<code>\ns@GLSxtrlong</code>	105, 106		
<code>\ns@glstrlong</code>	105		
<code>\ns@GLSxtrlongpl</code>	109		
<code>\ns@GLSxtrlongpl</code>	108		
<code>\ns@glstrlongpl</code>	108		
<code>\ns@GLSxtrshort</code>	105		
<code>\ns@GLSxtrshort</code>	104		
<code>\ns@glstrshort</code>	103, 104		
<code>\ns@GLSxtrshortpl</code>	107		
<code>\ns@GLSxtrshortpl</code>	107		
		O	
		<code>\or</code>	5, 8
		<code>\org@glossaryentrynumbers</code>	18
		P	
		<code>\p@glshyp@opt</code>	33
		package options:	
		abbreviations	6
		accsupp	9, 63
		acronym	6
		automake	56, 60
		docdef	13, 40, 48
		nonumberlist	18
		numbers	7
		shortcuts	8
		all	8
		false	8
		none	8
		true	8
		symbols	7, 80
		undefaction	11
		warn	5
		<code>\PackageError</code>	5, 14,
		17, 29–31, 39–41, 48, 50, 51, 53–56, 112, 113
		<code>\PackageWarning</code>	6
		<code>\PackageWarningNoLine</code>	6
		<code>\par</code>	61, 62
		<code>\PassOptionsToPackage</code>	4
		<code>\preto</code>	32
		<code>\printabbreviations</code>	6
		<code>\printglossaries</code>	55, 61
		<code>\printglossary</code>	6, 55, 61
		<code>\printglossary options</code>	
		nonumberlist	18
		<code>\printnoidxglossaries</code>	61
		<code>\printnoidxglossary</code>	61
		<code>\printnumbers</code>	7, 80
		<code>\printsymbols</code>	7, 80
		<code>\ProcessOptionsX</code>	9
		<code>\protect</code>	73–76, 99, 114–130, 135–139
		<code>\protected@edef</code>	52, 86, 98
		<code>\protected@write</code>	54
		<code>\providecommand</code>	6, 31, 41, 50, 54, 58, 59
		<code>\ProvidesFile</code>	143
		<code>\ProvidesPackage</code>	4

Q		T	
<code>\quotechar</code>	89	<code>\TeX</code>	60
R		<code>\texorpdfstring</code>	140–143
<code>\relax</code>	5, 7–10, 13, 17, 18, 33, 41, 42, 50, 55, 57, 87, 89, 91, 93, 96, 97	<code>textcase package</code>	133
<code>relsize package</code>	127	<code>\textsc</code>	124
<code>\renewcommand</code>	5, 6, 8, 9, 11–14, 17–19, 29, 31, 32, 34, 38–41, 45, 48–56, 58, 59, 80–85, 89–93, 100, 113–133	<code>\textsmaller</code>	127
<code>\RequireGlossariesExtraLang</code>	144	<code>\texttt</code>	59–62
<code>\RequirePackage</code>	4, 9	<code>\the</code>	52, 89, 98, 114–124
<code>\reserved@a</code>	95	<code>\theindex</code>	85
<code>\reserved@b</code>	95	<code>\this@dialect</code>	144
<code>\reserved@d</code>	95	<code>\toks@</code>	89
<code>\RestoreAcronyms</code>	53	U	
S		<code>\undef</code>	90
<code>\s@gl@s@hyp@opt</code>	33	<code>\underline</code>	92
<code>\s@gl@xtr@enabletagging</code>	90	<code>\unskip</code>	13
<code>\setabbreviationstyle</code>	53, 115, 121	<code>\usepackage</code>	61, 62
<code>\setacronymstyle</code>	53	V	
<code>\SetGenericNewAcronym</code>	53	<code>\val</code>	5, 8, 9
<code>\setkeys</code>	10, 32, 52, 97, 98	W	
<code>\setlength</code>	17, 18	<code>\warn@nomakeglossaries</code>	55
<code>\settowidth</code>	54	<code>\warn@noprintglossary</code>	55
<code>\setupglossaries</code>	4, 9	<code>\write</code>	41, 50, 54, 58, 59
<code>\sfcode</code>	93	X	
<code>\space</code>	5, 14, 17, 39–42, 48, 50, 51, 53–55, 59, 62, 93, 94, 99	<code>\xcapitalisewords</code>	81
<code>\spacefactor</code>	93, 97	<code>\xifinlist</code>	46
<code>\string</code>	5, 14, 17, 29, 30, 39–42, 48, 50, 51, 53–56, 58–62, 86	<code>xindy</code>	54
		<code>xkeyval package</code>	4
		<code>\XKV@checkchoice</code>	18
		<code>\XKV@plfalse</code>	18
		<code>\XKV@resa</code>	18, 19
		<code>\XKV@sttrue</code>	18