

`glossaries-extra.sty v1.02: documented code`

Nicola L.C. Talbot

Dickimaw Books

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

2016-04-25

Abstract

This is the documented code for the glossaries-extra package. See [glossaries-extra-manual.pdf](#) for the user manual.

This package is experimental and not stable. It's provided for testing purposes only. Future versions may not be compatible with this version. Once it has stabilised I'll add it to CTAN, at which point compatibility with the first stable version will be maintained.

Contents

1	Main Package Code (glossaries-extra.sty)	4
1.1	Package Initialisation and Options	4
1.2	Extra Utilities	11
1.3	Modifications to Commands Provided by glossaries	11
1.3.1	Existence Checks	11
1.3.2	Document Definitions	14
1.3.3	Existing Glossary Style Modifications	18
1.3.4	Entry Formatting, Hyperlinks and Indexing	20
1.3.5	Entry Counting	40
1.3.6	Acronym Modifications	53
1.3.7	Indexing and Displaying Glossaries	55
1.4	Integration with glossaries-accsupp	64
1.5	Categories	78
1.6	Abbreviations	98
1.6.1	Abbreviation Styles Setup	114
1.6.2	Predefined Styles (Default Font)	116
1.6.3	Predefined Styles (Small Capitals)	127
1.6.4	Predefined Styles (Fake Small Capitals)	130
1.6.5	Predefined Styles (Emphasized)	133
1.7	Using Entries in Headings	136
1.8	Multi-Lingual Support	153
2	Style Adjustments (glossaries-extra-stylemods.sty)	154
2.1	Package Initialisation	154
2.2	List-Like Styles	155
2.3	Longtable Styles	155
2.4	Long Ragged Styles	156
2.5	Supertabular Styles	158
2.6	Super Ragged Styles	159
2.7	Inline Style	160
	Glossary	161
	Change History	162
	Index	167

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2016/04/25 v1.02 (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 }
```

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

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

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

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

This text is switched on at the start of the document to prevent unwanted text inserted into the preamble if any tests are made before the start of the document.

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

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]{}
```

`\indexcrossrefs` 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 \@glsxtrindexcrossrefsfalse
```

But allow see key to switch it on automatically.

oindexcrossrefs

```
64 \newcommand*{\@glsxtr@autoindexcrossrefs}{\@glsxtrindexcrossrefstrue}
```

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 \@glsxtr@declareoption{nowarn}{%
69 \let\GlossariesExtraWarning\@gobble
70 \let\GlossariesExtraWarningNoLine\@gobble
71 \glsxtr@doption{nowarn}%
72 }
```

glsxtrabbrvtype

Glossary type for abbreviations.

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

bbreviationsdef

Set by abbreviations option.

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

bbreviationsdef

```
75 \newcommand*{\@glsxtr@doabbreviationsdef}{%
76 \@ifpackageloaded{babel}%
77 {\providecommand{\abbreviationsname}{\acronymname}}%
78 {\providecommand{\abbreviationsname}{Abbreviations}}%
79 \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
80 \renewcommand*{\glsxtrabbrvtype}{abbreviations}%
81 \newcommand*{\printabbreviations}[1][1]{%
82 \printglossary[type=\glsxtrabbrvtype,##1]%
83 }%
84 \disable@keys{glossaries-extra.sty}{abbreviations}%
```

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

```
85 \ifglsacronym
86 \else
87 \renewcommand*{\acronymtype}{\glsxtrabbrvtype}%
88 \fi
89 }%
```

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

```

90 \@glxtr@declareoption{abbreviations}{%
91   \let\@glxtr@abbreviationsdef\@glxtr@doabbreviationsdef
92 }

```

AbbreviationShortcuts 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{\cgl}%
95   \newcommand*\abp{\cglsp}%
96   \newcommand*\as{\glxtrshort}%
97   \newcommand*\asp{\glxtrshortpl}%
98   \newcommand*\al{\glxtrlong}%
99   \newcommand*\alp{\glxtrlongpl}%
100  \newcommand*\af{\glxtrfull}%
101  \newcommand*\afp{\glxtrfullpl}%
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{\cGLSp}%
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 }

```

OtherShortcuts 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{\glxtrnewsymbol}%
126     }{}%
127   \ifdef\printnumbers
128     {%
129       \newcommand*\newnum{\glxtrnewnumber}%

```

```

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 }

nomissingglsstext  If true, suppress the text produced if the external glossary file is missing.
182 \define@choicekey{glossaries-extra.sty}{nomissingglsstext}[\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 }

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

\lsxtr@redefstyles
194 \newcommand*{\@glxtr@redefstyles}{}

stylemods
195 \define@key{glossaries-extra.sty}{stylemods}{%
196   \ifblank{#1}%
197   {%
198     \renewcommand*{\@glxtr@redefstyles}{%
199       \RequirePackage{glossaries-extra-stylemods}}%
200   }%
201   {%

```

```

202 \renewcommand*{\@glxtr@redefstyles}{}%
203 \for\@glxtr@tmp:=#1\do{%
204 \IfFileExists{glossary-\@glxtr@tmp.sty}%
205 {%
206 \eappto\@glxtr@redefstyles{%
207 \noexpand\RequirePackage{glossary-\@glxtr@tmp}}%
208 }%
209 {%
210 \PackageError{glossaries-extra}%
211 {Glossaries style package ‘glossary-\@glxtr@tmp.sty’
212 doesn’t exist (did you mean to use the ‘style’ key?)}%
213 {The list of values (#1) in the ‘styles’ key should
214 match the glossary-xxx.sty files provided with
215 glossaries.sty}%
216 }%
217 }%
218 \appto\@glxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
219 }%
220 }

```

Pass all other options to glossaries.

```

221 \DeclareOptionX*{%
222 \expandafter\glxtr@doption\expandafter{\CurrentOption}}

```

Process options.

```
223 \ProcessOptionsX
```

Load glossaries if not already loaded.

```
224 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
225 \@glxtr@doaccsupp
```

Define abbreviations glossaries if required.

```

226 \@glxtr@abbreviationsdef
227 \let\@glxtr@abbreviationsdef\relax

```

Setup shortcuts if required.

```
228 \@glxtr@setupshortcuts
```

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

```
229 \renewcommand{\glxtr@doption}[1]{\setupglossaries{#1}}%
```

Now define the user command:

```

230 \newcommand*\glossariesextrasetup[1]{%
231 \let\@glxtr@setupshortcuts\relax
232 \setkeys{glossaries-extra.sty}{#1}%
233 \@glxtr@abbreviationsdef
234 \let\@glxtr@abbreviationsdef\relax
235 \@glxtr@setupshortcuts
236 }

```

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

```
237 \AtBeginDocument{%
238   \disable@keys{glossaries-extra.sty}{abbreviations}%
239   \def\@glxtrundeftag{\glxtrundeftag}%
240 }
```

1.2 Extra Utilities

`\glxtrifemptyglossary`

```
\glxtrifemptyglossary{<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.)

```
241 \newcommand{\glxtrifemptyglossary}[3]{%
242   \ifglossaryexists{#1}%
243   {%
244     \ifcsstring{glolist@#1}{,}{#2}{#3}%
245   }%
246   {%
247     \glxtrundefaction{Glossary type '#1' doesn't exist}{}%
248     #2%
249   }%
250 }
```

1.3 Modifications to Commands Provided by glossaries

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

1.3.1 Existence Checks

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

```
251 \renewcommand{\glstoifexists}[2]{%
252   \ifglstryexists{#1}{#2}%
253   {%
254     \glxtrundefaction{Glossary entry '\glstetoklabel{#1}'
255       has not been defined}{You need to define a glossary entry before
256       you can reference it.}%
257   }%
258 }
```

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

```
259 \renewcommand{\glsdoifnoexists}[2]{%
260   \ifglstryexists{#1}{%
261     \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
262       has already been defined}{}}{#2}%
263 }
```

`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.

```
264 \ifdef\glsdoifexistsordo
265 {%
266   \renewcommand{\glsdoifexistsordo}[3]{%
267     \ifglstryexists{#1}{#2}%
268     {%
269       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
270         has not been defined}{You need to define a glossary entry
271         before you can use it.}%
272       #3%
273     }%
274   }%
275 }
276 {%
277   \glstr@warnonexistsordo\glsdoifexistsordo
278   \newcommand{\glsdoifexistsordo}[3]{%
279     \ifglstryexists{#1}{#2}%
280     {%
281       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
282         has not been defined}{You need to define a glossary entry
283         before you can use it.}%
284       #3%
285     }%
286   }%
287 }
```

`glsdoifnoexistsordo` Similarly for `\doifglossarynoexistsordo`.

```
288 \ifdef\doifglossarynoexistsordo
289 {%
290   \renewcommand{\doifglossarynoexistsordo}[3]{%
291     \ifglossaryexists{#1}%
292     {%
293       \glstrundefaction{Glossary type '#1' already exists}{}%
294       #3%
295     }%
296     {#2}%
297   }%
298 }
299 {%
300   \glstr@warnonexistsordo\doifglossarynoexistsordo
301   \newcommand{\doifglossarynoexistsordo}[3]{%

```

```

302   \ifglossaryexists{#1}%
303   {%
304     \glstrundefaction{Glossary type ‘#1’ already exists}{}%
305     #3%
306   }%
307   {#2}%
308 }%
309 }
310

```

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

```

311 \appto\@newglossaryentryposthook{%
312   \ifdefvoid\@glo@see
313   {\csxdef{glo@\@glo@label @see}{}}%
314   {%
315     \csxdef{glo@\@glo@label @see}{\@glo@see}%
316     \@glstr@autoindexcrossrefs
317   }%
318 }
319 \appto\@gls@keymap{,{see}{see}}

```

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

```

320 \AtEndDocument{\if@glstr@indexcrossrefs\glstr@addallcrossrefs\fi}

```

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

```

321 \newcommand*\glstr@addallcrossrefs{%
322   \forallglossaries{\@glo@type}%
323   {%
324     \forallglsentries[\@glo@type]{\@glo@label}%
325     {%
326       \ifglsused{\@glo@label}{\@glstr@addunusedxrefs{\@glo@label}}{}%
327     }%
328   }%
329 }

```

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

```

330 \newcommand*\@glstr@addunusedxrefs[1]{%
331   \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
332   \ifdefvoid\@glo@see
333   {}%
334   {%
335     \expandafter\glstr@addunused\@glo@see\@end@glstr@addunused
336   }%
337 }

```

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

```

338 \newcommand*\glstr@addunused[1][]{%

```

```

339 \@glxstr@addunused
340 }

```

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

```

341 \def\@glxstr@addunused#1\@end@glxstr@addunused{%
342 \@for\@glxstr@label:=#1\do
343 {%
344 \ifglxsused{\@glxstr@label}{}%
345 {%
346 \glsadd[format=glxstrunusedformat]{\@glxstr@label}%
347 \glsunset{\@glxstr@label}%
348 \@glxstr@addunusedxrefs{\@glxstr@label}%
349 }%
350 }%
351 }

```

`\glxstrunusedformat`

```

352 \newcommand*{\glxstrunusedformat}[1]{\unskip}

```

1.3.2 Document Definitions

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

```

353 \let\glxstr@orgmakenoidxglossaries\makenoidxglossaries
354 \renewcommand{\makenoidxglossaries}{%
355 \glxstr@orgmakenoidxglossaries
356 \@glxstrdocdeffalse
357 \disable@keys{glossaries-extra.sty}{docdef}%
358 }

```

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

```

359 \renewcommand*{\gls@defdocnewglossaryentry}{%
360 \if@glxstrdocdef

```

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

```

361 \let\gls@checkseeallowed\relax
362 \let\newglossaryentry\new@glossaryentry
363 \else
364 \renewcommand*{\newglossaryentry}[2]{%
365 \PackageError{glossaries-extra}{Glossary entries must
366 be \MessageBreak defined in the preamble with \MessageBreak
367 package option 'docdef=false'}{Move your glossary definitions to
368 the preamble. You can also put them in a \MessageBreak separate file
369 and load them with \string\loadglsentries.}%
370 }%
371 \fi
372 }%

```

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`

```
373 \newcommand*{\GlsXtrEnableOnTheFly}{%
374   \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
375 }
```

`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.)

```
376 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
377   \renewcommand*{\glsdetoklabel}[1]{%
378     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
379     {%
380       \expandafter\detokenize\expandafter{##1}%
381     }%
382     {\detokenize{##1}}}%
383   }%
384   \@GlsXtrEnableOnTheFly
385 }
386 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
387   \expandafter\if\glsbackslash#1%
388     #3%
389   \else
390     #4%
391   \fi
392 }
```

`sxtrstarflywarn`

```
393 \newcommand*{\glsxtrstarflywarn}{%
394   \GlossariesExtraWarning{Experimental starred version of
395   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
396   read the warnings in the glossaries-extra user manual)}}%
397 }
```

`rEnableOnTheFly`

```
398 \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`

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

```

\glxstr
400 \newcommand*\glxstr}[1] [] {%
401 \def\glxstr@keylist{##1}%
402 \@glxstr
403 }

\@glxstr
404 \newcommand*\@glxstr}[2] [] {%
405 \ifglentryexists{##2}%
406 {%
407 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
408 }%
409 {%
410 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
411 description={\nopostdesc},##1}%
412 }%
413 \expandafter\gls\expandafter[\glxstr@keylist]{##2}%
414 }

\Glsxtr
415 \newcommand*\Glsxtr}[1] [] {%
416 \def\glxstr@keylist{##1}%
417 \@Glsxtr
418 }

\@Glsxtr
419 \newcommand*\@Glsxtr}[2] [] {%
420 \ifglentryexists{##2}%
421 {%
422 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
423 }%
424 {%
425 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
426 description={\nopostdesc},##1}%
427 }%
428 \expandafter\Gls\expandafter[\glxstr@keylist]{##2}%
429 }

\glxstrpl
430 \newcommand*\glxstrpl}[1] [] {%
431 \def\glxstr@keylist{##1}%
432 \@glxstrpl
433 }

\@glxstrpl
434 \newcommand*\@glxstrpl}[2] [] {%
435 \ifglentryexists{##2}%
436 {%

```



```

437     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
438 }%
439 {%
440     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
441     description={\nopostdesc},##1}%
442 }%
443 \expandafter\glspl\expandafter[\glsxtr@keylist]{##2}%
444 }

```

\Glsxtrpl

```

445 \newcommand*\Glsxtrpl[1][1]{%
446 \def\glsxtr@keylist{##1}%
447 \@Glsxtrpl
448 }

```

\@Glsxtrpl

```

449 \newcommand*\@Glsxtrpl[2][1]{%
450 \ifglsentryexists{##2}
451 {%
452     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
453 }%
454 {%
455     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
456     description={\nopostdesc},##1}%
457 }%
458 \expandafter\Glspl\expandafter[\glsxtr@keylist]{##2}%
459 }

```

\GlsXtrWarning

```

460 \newcommand*\GlsXtrWarning[2]{%
461 \def\glsxtr@optlist{##1}%
462 \@onelevel@sanitize\glsxtr@optlist
463 \GlossariesExtraWarning{The options '\glsxtr@optlist' have
464 been ignored for entry '##2' as it has already been defined}%
465 }

```

Disable commands after the glossary:

```

466 \let\glsxtr@orgprintglossary\@printglossary
467 \renewcommand\@printglossary[2]{%
468 \glsxtr@orgprintglossary{##1}{##2}%
469 \def\glsxtr{\glsxtr@disabledflycommand\glsxtr}%
470 \def\glsxtrpl{\glsxtr@disabledflycommand\glsxtrpl}%
471 \def\Glsxtr{\glsxtr@disabledflycommand\Glsxtr}%
472 \def\Glsxtrpl{\glsxtr@disabledflycommand\Glsxtrpl}%
473 }

```

\glsxtr@disabledflycommand

```

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

```

```

476   {\string##1\space can't be used after any of the \MessageBreak
477     glossaries have been displayed}%
478   {The on-the-fly commands enabled by
479     \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
480     before the glossaries. If you want to use any entries \MessageBreak
481     after any of the glossaries, you must use the standard \MessageBreak
482     method of first defining the entry and then using the \MessageBreak
483     entry with commands like \string\gls}%
484     @@glxtr@disabledflycommand
485   }%
486   \newcommand*{@@glxtr@disabledflycommand}[2][\{##2}

```

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

```

487   \let\GlsXtrEnableOnTheFly\relax
488 }
489 \@onlypreamble\GlsXtrEnableOnTheFly

```

1.3.3 Existing Glossary Style Modifications

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

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

```

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

```

Modify \setglossarystyle to set the above.

etglossarystyle

```

491 \renewcommand*{\setglossarystyle}[1]{%
492   \ifcsundef{@glsstyle@#1}%
493   {%
494     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
495   }%
496   {%
497     \csname @glsstyle@#1\endcsname

```

Only set the current style if it exists.

```

498     \protected@edef\@glxtr@current@style{#1}%
499   }%
500   \ifx\@glossary@default@style\relax
501     \protected@edef\@glossary@default@style{#1}%
502   \fi
503 }

```

In case we have an old version of glossaries:

```

504 \ifdef\@glossary@default@style
505 {}
506 {%
507   \let\@glossary@default@style\relax
508 }

```

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

```

509 \ifdef\glslistdottedwidth
510 {%
511   \ifdim\glslistdottedwidth=.5\hsize
512     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
513     \AtBeginDocument{%
514       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
515         \setlength{\glslistdottedwidth}{.5\columnwidth}%
516       \fi
517     }%
518   \fi
519 }
520 {}%
```

Similarly for \glsdescwidth:

\glsdescwidth

```

521 \ifdef\glsdescwidth
522 {%
523   \ifdim\glsdescwidth=.6\hsize
524     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
525     \AtBeginDocument{%
526       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
527         \setlength{\glsdescwidth}{.6\columnwidth}%
528       \fi
529     }%
530   \fi
531 }
532 {}%
```

and for \glspagelistwidth:

lspagelistwidth

```

533 \ifdef\glspagelistwidth
534 {%
535   \ifdim\glspagelistwidth=.1\hsize
536     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
537     \AtBeginDocument{%
538       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
539         \setlength{\glspagelistwidth}{.1\columnwidth}%
540       \fi
541     }%
542   \fi
543 }
544 {}%
```

aryentrynumbers Has the nonumberlist option been used?

```

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

```

546 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
547 \glsglossaryentrynumbersfalse
548 \renewcommand*\glsXtrFormatLocationList{#1}{\glsglossaryentrynumbers{#1}}%
549 \GlsXtrFormatLocationList{#1}\glsglossaryentrynumbers{#1}}%
550 \else
551 \glsglossaryentrynumberstrue
552 \renewcommand*\glsXtrFormatLocationList{#1}{\glsglossaryentrynumbers{#1}}%
553 \fi

```

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

```

554 \newcommand*\GlsXtrFormatLocationList{#1}{#1}

```

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

```

555 \renewcommand*\KV@printgloss@nonumberlist{#1}{%
556 \XKV@plfalse
557 \XKV@sttrue
558 \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
559 {%
560 \csname glsglossaryentrynumbers\XKV@resa\endcsname
561 \ifglsglossaryentrynumbers
562 \def\glsXtrFormatLocationList{#1}{\glsglossaryentrynumbers{#1}}%
563 \else
564 \def\glsXtrFormatLocationList{#1}{%
565 \GlsXtrFormatLocationList{#1}}%
566 \glsglossaryentrynumbers{#1}}%
567 \fi
568 }%
569 }

```

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.

```

570 \renewcommand*\glsentryfmt{#1}{%
571 \ifglsglossaryentrynumbers{#1}{\glsglossaryentrynumbers{#1}}%
572 \glsglossaryentrynumbers{#1}}%
573 {\glsglossaryentrynumbers{#1}}%
574 {\ifglsglossaryentrynumbers{#1}{\glsglossaryentrynumbers{#1}}%
575 }

```

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.

```
576 \renewcommand{\@gls@field@link}[4][]{%
577   \glsdoifexists{#3}%
578   {%
579     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
580     \def\glscustomtext{#4}%
581     \@glsxtr@field@linkdefs
582     #1%
583     \@gls@link[#2]{#3}{#4}%
584   }%
585   \glspostlinkhook
586 }
```

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

```
587 \newcommand*{\@glsxtr@field@linkdefs}{%
588   \let\glsxtrifwasfirstuse\@secondoftwo
589   \let\glsifplural\@secondoftwo
590   \let\glscapscase\@firstofthree
591   \let\glsinsert\@empty
592 }
```

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

\@GLStext@ All uppercase version of \gls text.

```
593 \def\@GLStext@#1#2[#3]{%
594   \@gls@field@link[\let\glsapscase\@thirdofthree]{#1}{#2}%
595   {\mfirstucMakeUppercase{\glsentrytext{#2}#3}}%
596 }
```

\@Glstext@ First letter uppercase version.

```
597 \def\@Glstext@#1#2[#3]{%
598   \@gls@field@link[\let\glsapscase\@secondofthree]{#1}{#2}%
599   {\Glentrytext{#2}#3}%
600 }
```

\@glsfirst@ No case changing version.

```
601 \def\@glsfirst@#1#2[#3]{%
602   \@gls@field@link[\let\glsxtrifwasfirstuse\@firstoftwo]{#1}{#2}%
603   {\glsentryfirst{#2}#3}%
604 }
```

\@Glsfirst@ First letter uppercase version.

```
605 \def\@Glsfirst@#1#2[#3]{%
```

```

606 \@gls@field@link
607 [\let\glstrifwasfirstuse\@firstoftwo
608 \let\glscapscase\@secondofthree
609 ]%
610 {#1}{#2}{\Glsentryfirst{#2}#3}%
611 }

```

\@GLSfirst@ All uppercase version.

```

612 \def\@GLSfirst@#1#2[#3]{%
613 \@gls@field@link
614 [\let\glstrifwasfirstuse\@firstoftwo
615 \let\glscapscase\@thirdofthree
616 ]%
617 {#1}{#2}{\mfirstucMakeUppercase{\glentryfirst{#2}#3}}%
618 }

```

\@glsplural No case changing version.

```

619 \def\@glsplural#1#2[#3]{%
620 \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
621 {\glentryplural{#2}#3}%
622 }

```

\@Glsplural First letter uppercase version.

```

623 \def\@Glsplural#1#2[#3]{%
624 \@gls@field@link
625 [\let\glsifplural\@firstoftwo
626 \let\glscapscase\@secondofthree
627 ]%
628 {#1}{#2}{\Glsentryplural{#2}#3}%
629 }

```

\@GLSplural All uppercase version.

```

630 \def\@GLSplural#1#2[#3]{%
631 \@gls@field@link
632 [\let\glsifplural\@firstoftwo
633 \let\glscapscase\@thirdofthree
634 ]%
635 {#1}{#2}{\mfirstucMakeUppercase{\glentryplural{#2}#3}}%
636 }

```

glsfirstplural@ No case changing version.

```

637 \def\glsfirstplural@#1#2[#3]{%
638 \@gls@field@link
639 [\let\glstrifwasfirstuse\@firstoftwo
640 \let\glsifplural\@firstoftwo
641 ]%
642 {#1}{#2}{\glentryfirstplural{#2}#3}%
643 }

```

Glsfirstplural@ First letter uppercase version.

```
644 \def\@glsfirstplural@#1#2[#3]{%
645   \@gls@field@link
646   [\let\glstrifwasfirstuse\@firstoftwo
647   \let\glsifplural\@firstoftwo
648   \let\glscapscase\@secondofthree
649   ]%
650   {#1}{#2}{\Glsentryfirstplural{#2}#3}%
651 }
```

Glsfirstplural@ All uppercase version.

```
652 \def\@glsfirstplural@#1#2[#3]{%
653   \@gls@field@link
654   [\let\glstrifwasfirstuse\@firstoftwo
655   \let\glsifplural\@firstoftwo
656   \let\glscapscase\@thirdofthree
657   ]%
658   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryfirstplural{#2}#3}}%
659 }
```

\@Glsname@ First letter uppercase version.

```
660 \def\@Glsname@#1#2[#3]{%
661   \@gls@field@link
662   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryname{#2}#3}%
663 }
```

\@GLSname@ All uppercase version.

```
664 \def\@GLSname@#1#2[#3]{%
665   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
666   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryname{#2}#3}}%
667 }
```

\@Glsdesc@ First letter uppercase version.

```
668 \def\@Glsdesc@#1#2[#3]{%
669   \@gls@field@link
670   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentrydesc{#2}#3}%
671 }
```

\@GLSdesc@ All uppercase version.

```
672 \def\@GLSdesc@#1#2[#3]{%
673   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
674   {#1}{#2}{\mfirstucMakeUppercase{\Glsentrydesc{#2}#3}}%
675 }
```

@glsdescplural@ No case-changing version.

```
676 \def\@glsdescplural@#1#2[#3]{%
677   \@gls@field@link
678   [\let\glscapscase\@secondoftwo
```

```

679 \let\glsifplural\@firstoftwo
680 ]{\#1}{\#2}{\glsentrydescplural{\#2}{\#3}}%
681 }

```

@Glsdescplural@ First letter uppercase version.

```

682 \def\@Glsdescplural@#1#2[#3]{%
683 \@gls@field@link
684 [\let\glscapscase\@secondoftwo
685 \let\glsifplural\@firstoftwo
686 ]{\#1}{\#2}{\Glsentrydescplural{\#2}{\#3}}%
687 }

```

@GLSdescplural@ All uppercase version.

```

688 \def\@GLSdesc@#1#2[#3]{%
689 \@gls@field@link
690 [\let\glscapscase\@thirdoftwo
691 \let\glsifplural\@firstoftwo
692 ]%
693 {\#1}{\#2}{\mfirstucMakeUppercase{\glsentrydescplural{\#2}{\#3}}}%
694 }

```

\@GLssymbol@ First letter uppercase version.

```

695 \def\@GLssymbol@#1#2[#3]{%
696 \@gls@field@link
697 [\let\glscapscase\@secondoftwo]{\#1}{\#2}{\Glsentrysymbol{\#2}{\#3}}%
698 }

```

\@GLSsymbol@ All uppercase version.

```

699 \def\@GLSsymbol@#1#2[#3]{%
700 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
701 {\#1}{\#2}{\mfirstucMakeUppercase{\glsentrysymbol{\#2}{\#3}}}%
702 }

```

lssymbolplural@ No case-changing version.

```

703 \def\@glssymbolplural@#1#2[#3]{%
704 \@gls@field@link
705 [\let\glscapscase\@secondoftwo
706 \let\glsifplural\@firstoftwo
707 ]{\#1}{\#2}{\glsentrysymbolplural{\#2}{\#3}}%
708 }

```

lssymbolplural@ First letter uppercase version.

```

709 \def\@GLssymbolplural@#1#2[#3]{%
710 \@gls@field@link
711 [\let\glscapscase\@secondoftwo
712 \let\glsifplural\@firstoftwo
713 ]{\#1}{\#2}{\Glsentrysymbolplural{\#2}{\#3}}%
714 }

```


LSSymbolplural@ All uppercase version.

```

715 \def\@GLSSymbol@#1#2[#3]{%
716   \@gls@field@link
717   [\let\glscapscase\@thirdoftwo
718     \let\glsifplural\@firstoftwo
719   ]%
720   {#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbolplural{#2}#3}}%
721 }

```

\@Glsuseri@ First letter uppercase version.

```

722 \def\@Glsuseri@#1#2[#3]{%
723   \@gls@field@link
724   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuseri{#2}#3}%
725 }

```

\@GLSuseri@ All uppercase version.

```

726 \def\@GLSuseri@#1#2[#3]{%
727   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
728   {#1}{#2}{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}%
729 }

```

\@Glsuserii@ First letter uppercase version.

```

730 \def\@Glsuserii@#1#2[#3]{%
731   \@gls@field@link
732   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuserii{#2}#3}%
733 }

```

\@GLSuserii@ All uppercase version.

```

734 \def\@GLSuserii@#1#2[#3]{%
735   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
736   {#1}{#2}{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}%
737 }

```

\@Glsuseriii@ First letter uppercase version.

```

738 \def\@Glsuseriii@#1#2[#3]{%
739   \@gls@field@link
740   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuseriii{#2}#3}%
741 }

```

\@GLSuseriii@ All uppercase version.

```

742 \def\@GLSuseriii@#1#2[#3]{%
743   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
744   {#1}{#2}{\mfirstucMakeUppercase{\glsentryuseriii{#2}#3}}%
745 }

```

\@Glsuseriv@ First letter uppercase version.

```

746 \def\@Glsuseriv@#1#2[#3]{%
747   \@gls@field@link

```

```

748 [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuseriv{#2}#3}%
749 }

```

\@GLSuseriv@ All uppercase version.

```

750 \def\@GLSuseriv#1#2[#3]{%
751   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
752   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryuseriv{#2}#3}}%
753 }

```

\@Glsuserv@ First letter uppercase version.

```

754 \def\@Glsuserv#1#2[#3]{%
755   \@gls@field@link
756   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuserv{#2}#3}%
757 }

```

\@GLSuserv@ All uppercase version.

```

758 \def\@GLSuserv#1#2[#3]{%
759   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
760   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryuserv{#2}#3}}%
761 }

```

\@Glsuservi@ First letter uppercase version.

```

762 \def\@Glsuservi#1#2[#3]{%
763   \@gls@field@link
764   [\let\glscapscase\@secondoftwo]{#1}{#2}{\Glsentryuservi{#2}#3}%
765 }

```

\@GLSuservi@ All uppercase version.

```

766 \def\@GLSuservi#1#2[#3]{%
767   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
768   {#1}{#2}{\mfirstucMakeUppercase{\Glsentryuservi{#2}#3}}%
769 }

```

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

\@acrshort No case change.

```

770 \def\@acrshort#1#2[#3]{%
771   \glsdoifexists{#2}%
772   {%
773     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
774     \let\glxtrifwasfirstuse\@secondoftwo
775     \let\glsifplural\@secondoftwo
776     \let\glscapscase\@firstofthree
777     \let\glsinsert\@empty
778     \def\glscustomtext{%
779       \acronymfont{\glsaccessshort{#2}}#3%
780     }%

```

```

781 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
782 }%
783 \glspostlinkhook
784 }

```

\@Acrshort First letter uppercase.

```

785 \def\@Acrshort#1#2[#3]{%
786 \glsdoifexists{#2}%
787 {%
788 \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
789 \let\glstrifwasfirstuse\@secondoftwo
790 \let\gl@sifplural\@secondoftwo
791 \let\glscapscase\@secondofthree
792 \let\gl@sinsert\@empty
793 \def\glscustomtext{%
794 \acronymfont{\Glsaccessshort{#2}}#3%
795 }%
796 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
797 }%
798 \glspostlinkhook
799 }

```

\@ACRshort All uppercase.

```

800 \def\@ACRshort#1#2[#3]{%
801 \glsdoifexists{#2}%
802 {%
803 \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
804 \let\glstrifwasfirstuse\@secondoftwo
805 \let\gl@sifplural\@secondoftwo
806 \let\glscapscase\@thirdofthree
807 \let\gl@sinsert\@empty
808 \def\glscustomtext{%
809 \mfirstucMakeUppercase{\acronymfont{\Glsaccessshort{#2}}#3}%
810 }%
811 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
812 }%
813 \glspostlinkhook
814 }

```

\@acrshortpl No case change.

```

815 \def\@acrshortpl#1#2[#3]{%
816 \glsdoifexists{#2}%
817 {%
818 \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
819 \let\glstrifwasfirstuse\@secondoftwo
820 \let\gl@sifplural\@firstoftwo
821 \let\glscapscase\@firstofthree
822 \let\gl@sinsert\@empty
823 \def\glscustomtext{%

```

```

824     \acronymfont{\glsaccesssshortpl{#2}}#3%
825   }%
826   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
827 }%
828 \glspostlinkhook
829 }

```

\@Acrshortpl First letter uppercase.

```

830 \def\@Acrshortpl#1#2[#3]{%
831   \glsdoifexists{#2}%
832   {%
833     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
834     \let\glstrifwasfirstuse\@secondoftwo
835     \let\glsifplural\@firstoftwo
836     \let\glscapscase\@secondofthree
837     \let\glsinsert\@empty
838     \def\glscustomtext{%
839       \acronymfont{\Glsaccesssshortpl{#2}}#3%
840     }%
841     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
842   }%
843   \glspostlinkhook
844 }

```

\@ACRshortpl All uppercase.

```

845 \def\@ACRshortpl#1#2[#3]{%
846   \glsdoifexists{#2}%
847   {%
848     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
849     \let\glstrifwasfirstuse\@secondoftwo
850     \let\glsifplural\@firstoftwo
851     \let\glscapscase\@thirdofthree
852     \let\glsinsert\@empty
853     \def\glscustomtext{%
854       \mfirstucMakeUppercase{\acronymfont{\glsaccesssshortpl{#2}}#3}%
855     }%
856     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
857   }%
858   \glspostlinkhook
859 }

```

\@acrlong No case change.

```

860 \def\@acrlong#1#2[#3]{%
861   \glsdoifexists{#2}%
862   {%
863     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
864     \let\glstrifwasfirstuse\@secondoftwo
865     \let\glsifplural\@secondoftwo
866     \let\glscapscase\@firstofthree

```

```

867 \let\glsinsert\@empty
868 \def\glscustomtext{%
869 \acronymfont{\glsaccesslong{#2}}#3%
870 }%
871 \@gls@link[#1]{#2}{\cname gls@\glstype @entryfmt\endcsname}%
872 }%
873 \glspostlinkhook
874 }

```

\@Acrlong First letter uppercase.

```

875 \def\@Acrlong#1#2[#3]{%
876 \glsdoifexists{#2}%
877 {%
878 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
879 \let\glstrifwasfirstuse\@secondoftwo
880 \let\glsifplural\@secondoftwo
881 \let\glscapscase\@secondofthree
882 \let\glsinsert\@empty
883 \def\glscustomtext{%
884 \acronymfont{\Glsaccesslong{#2}}#3%
885 }%
886 \@gls@link[#1]{#2}{\cname gls@\glstype @entryfmt\endcsname}%
887 }%
888 \glspostlinkhook
889 }

```

\@ACRlong All uppercase.

```

890 \def\@ACRlong#1#2[#3]{%
891 \glsdoifexists{#2}%
892 {%
893 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
894 \let\glstrifwasfirstuse\@secondoftwo
895 \let\glsifplural\@secondoftwo
896 \let\glscapscase\@thirdofthree
897 \let\glsinsert\@empty
898 \def\glscustomtext{%
899 \mfirstucMakeUppercase{\acronymfont{\glsaccesslong{#2}}#3}%
900 }%
901 \@gls@link[#1]{#2}{\cname gls@\glstype @entryfmt\endcsname}%
902 }%
903 \glspostlinkhook
904 }

```

\@acrlongpl No case change.

```

905 \def\@acrlongpl#1#2[#3]{%
906 \glsdoifexists{#2}%
907 {%
908 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
909 \let\glstrifwasfirstuse\@secondoftwo

```

```

910 \let\glsifplural\@firstoftwo
911 \let\glsifcaps\@firstofthree
912 \let\glsinsert\@empty
913 \def\glscustomtext{%
914   \acronymfont{\glsaccesslongpl{#2}}#3%
915 }%
916 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
917 }%
918 \glspostlinkhook
919 }

```

`\@Acrlongpl` First letter uppercase.

```

920 \def\@Acrlongpl#1#2[#3]{%
921   \glsdoifexists{#2}%
922   {%
923     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
924     \let\glsxtrifwasfirstuse\@secondoftwo
925     \let\glsifplural\@firstoftwo
926     \let\glsifcaps\@secondofthree
927     \let\glsinsert\@empty
928     \def\glscustomtext{%
929       \acronymfont{\glsaccesslongpl{#2}}#3%
930     }%
931     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
932   }%
933   \glspostlinkhook
934 }

```

`\@ACRlongpl` All uppercase.

```

935 \def\@ACRlongpl#1#2[#3]{%
936   \glsdoifexists{#2}%
937   {%
938     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
939     \let\glsxtrifwasfirstuse\@secondoftwo
940     \let\glsifplural\@firstoftwo
941     \let\glsifcaps\@thirdofthree
942     \let\glsinsert\@empty
943     \def\glscustomtext{%
944       \mfirstucMakeUppercase{\acronymfont{\glsaccesslongpl{#2}}#3}%
945     }%
946     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
947   }%
948   \glspostlinkhook
949 }

```

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

`\@glsaddkey`

```

950 \renewcommand*{\@glsaddkey}[7]{%
951   \key@ifundefined{glossentry}{#1}%
952   {%
953     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
954     \appto\@gls@keymap{,{#1}{#1}}%
955     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
956     \appto\@newglossaryentryposthook{%
957       \letcs{\@glo@tmp}{@glo@#1}%
958       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
959     }%
960     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
961     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

962   \ifcsdef{@gls@user@#1@}%
963   {%
964     \PackageError{glossaries}%
965     {Can't define '\string#5' as helper command
966     '\expandafter\string\csname @gls@user@#1@endcsname' already
967     exists}%
968     {}%
969   }%
970   {%
971     \expandafter\newcommand\expandafter*\expandafter
972     {\csname @gls@user@#1@endcsname}[2][ ]{%
973       \new@ifnextchar[%
974         {\csuse{@gls@user@#1@}{##1}{##2}}%
975         {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
976     \csdef{@gls@user@#1@}##1##2[##3]{%
977       \@gls@field@link{##1}{##2}{#3{##2}##3}%
978     }%
979     \newrobustcmd*{#5}{%
980       \expandafter\@gls@hyp@opt\csname @gls@user@#1@endcsname}%
981   }%

```

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

```

982   \ifcsdef{@Gls@user@#1@}%
983   {%
984     \PackageError{glossaries}%
985     {Can't define '\string#6' as helper command
986     '\expandafter\string\csname @Gls@user@#1@endcsname' already
987     exists}%
988     {}%
989   }%
990   {%
991     \expandafter\newcommand\expandafter*\expandafter
992     {\csname @Gls@user@#1@endcsname}[2][ ]{%
993       \new@ifnextchar[%
994         {\csuse{@Gls@user@#1@}{##1}{##2}}%
995         {\csuse{@Gls@user@#1@}{##1}{##2}[ ]}}%

```

```

996      \csdef{@GLs@user@#1@}##1##2[##3]{%
997      \@gls@field@link[\let\glscapscase\@secondofthree]%
998      {##1}{##2}{#4{##2}##3}%
999      }%
1000     \newrobustcmd*{#6}{%
1001     \expandafter\@gls@hyp@opt\csname @GLs@user@#1\endcsname}%
1002     }%

```

Finally the all caps version (modified):

```

1003     \ifcsdef{@GLS@user@#1@}%
1004     {%
1005       \PackageError{glossaries}%
1006       {Can't define '\string#7' as helper command
1007       '\expandafter\string\csname @GLS@user@#1\endcsname' already
1008       exists}%
1009     }%
1010     }%
1011     {%
1012       \expandafter\newcommand\expandafter*\expandafter
1013       {\csname @GLS@user@#1\endcsname}[2][ ]{%
1014         \new@ifnextchar[%
1015         {\csuse{@GLS@user@#1@}{##1}{##2}}%
1016         {\csuse{@GLS@user@#1@}{##1}{##2}[ ]}}%
1017       \csdef{@GLS@user@#1@}##1##2[##3]{%
1018       \@gls@field@link[\let\glscapscase\@thirdofthree]%
1019       {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1020       }%
1021       \newrobustcmd*{#7}{%
1022       \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
1023       }%
1024     }%
1025     {%
1026       \PackageError{glossaries-extra}{Key '#1' already exists}{}%
1027     }%
1028 }

```

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

```

1029 \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.

```

1030 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
1031 \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.

```

1032 \ifglsused{\glslabel}%

```



```

1033 {\let\glxstrifwasfirstuse\@secondoftwo}
1034 {\let\glxstrifwasfirstuse\@firstoftwo}%

Store the category label for convenience.
1035 \edef\glscategorylabel{\glscategory{\glslabel}}%
1036 \ifglused{\glslabel}%
1037 {%
1038   \glscategoryattribute{\glscategorylabel}{nohypernext}{true}%
1039   {\KV@glslink@hyperfalse}{}}%
1040 }%
1041 {%
1042   \glscategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
1043   {\KV@glslink@hyperfalse}{}}%
1044 }%
1045 \glslinkcheckfirsthyperhook
1046 }

```

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.

```

1047 \ifdef\do@gl:disablehyperinlist
1048 {%
1049   \let\@glxstr\do@gl:disablehyperinlist\do@gl:disablehyperinlist
1050   \renewcommand*{\do@gl:disablehyperinlist}{%
1051     \@glxstr\do@gl:disablehyperinlist
1052     \glscategoryattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}}%
1053   }
1054 }
1055 {}

```

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

```

1056 \define@boolkey{glslink}{noindex}[true]{}
1057 \KV@glslink@noindexfalse

```

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

lt@glslink@opts

```

1058 \ifdef\@gl@setdefault@glslink@opts
1059 {
1060   \renewcommand*{\@gl@setdefault@glslink@opts}{%
1061     \KV@glslink@noindexfalse
1062   }
1063 }
1064 {

```

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

```

1065   \newcommand*{\@gl@setdefault@glslink@opts}{%
1066     \KV@glslink@noindexfalse
1067   }
1068   \pretol\do@gl:disablehyperinlist{\@gl@setdefault@glslink@opts}
1069 }

```

DefaultGlsOpts Set the default options for \glslink etc.

```
1070 \newcommand*\GlsXtrSetDefaultGlsOpts}[1]{%
1071   \renewcommand*\@gls@setdefault@glslink@opts{\setkeys{glslink}{#1}}%
1072 }
```

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

```
1073 \newcommand*\glsxtrifindexing}[2]{%
1074   \ifKV@glslink@noindex #2\else #1\fi
1075 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```
1076 \renewcommand*\glswriteentry}[2]{%
1077   \glsxtrifindexing
1078   {%
1079     \ifglsindexonlyfirst
1080       \ifglsused{#1}
1081       {\glsxtrdoautoindexname{#1}{dualindex}}%
1082       {#2}%
1083     \else
1084       \glsifattribute{#1}{indexonlyfirst}{true}%
1085       {\ifglsused{#1}
1086        {\glsxtrdoautoindexname{#1}{dualindex}}%
1087        {#2}}%
1088       {#2}%
1089     \fi
1090   }%
1091   {}%
1092 }
```

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

```
1093 \appto\@do@@wrglossary{\@glsxtr@do@@wrindex
1094   \glsxtrdowrglossaryhook{\@gls@label}}%
1095 }
```

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

Similarly for the “noidx” version:

s@noidxglossary

```
1096 \appto\gls@noidxglossary{\@glsxtr@do@@wrindex
1097   \glsxtrdowrglossaryhook{\@gls@label}}%
1098 }
```

xtr@do@@wrindex

```
1099 \newcommand*\@glsxtr@do@@wrindex{%
1100   \glsxtrdoautoindexname{\@gls@label}{dualindex}%
1101 }
```

`\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.)

```
1102 \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.

```
1103 \newcommand*{\@gls@alt@hyp@opt}[1]{%
1104   \let\glslinkvar\@firstofthree
1105   \let\@gls@hyp@opt@cs#1\relax
1106   \@ifstar{\s@gls@hyp@opt}%
1107   {\@ifnextchar+%
1108     {\@firstoftwo{\p@gls@hyp@opt}}%
1109     {%
1110       \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
1111       {\@firstoftwo{\@alt@gls@hyp@opt}}%
1112       {#1}%
1113     }%
1114   }%
1115 }
```

`\alt@gls@hyp@opt` User version

```
1116 \newcommand*{\@alt@gls@hyp@opt}[1][ ]{%
1117   \let\glslinkvar\@firstofthree
1118   \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}
```

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

```
1119 \newcommand*{\@gls@alt@hyp@opt@char}{{}
```

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

```
1120 \newcommand*{\@gls@alt@hyp@opt@keys}{{}
```

`\rSetAltModifier`

```
1121 \newcommand*{\GlsXtrSetAltModifier}[2]{%
1122   \let\@gls@hyp@opt\@gls@alt@hyp@opt
1123   \def\@gls@alt@hyp@opt@char{#1}%
1124   \def\@gls@alt@hyp@opt@keys{#2}%
1125 }
```

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

```
1126 \renewcommand*{\glsdohyperlink}[2]{%
1127   \hyperlink{#1}{{\glstrprotectlinks#2}}}
```

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

```
1128 \ifundef\glsdonohyperlink
1129 {%
1130   \renewcommand{\glsdisablehyper}{%
1131     \KV@glslink@hyperfalse
1132     \let\@glslink\glsdonohyperlink
1133     \let\@glstarget\@secondoftwo
1134   }
1135 }
1136 {}
```

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

```
1137 \def\glsdonohyperlink#1#2{{\glstrprotectlinks #2}}
```

Reset `\@glslink` with patched versions:

```
1138 \ifcsundef{hyperlink}%
1139 {%
1140   \let\@glslink\glsdonohyperlink
1141 }%
1142 {%
1143   \let\@glslink\glsdohyperlink
1144 }
```

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

```
1145 \newcommand*{\glstrprotectlinks}{%
1146   \KV@glslink@hyperfalse
1147   \KV@glslink@noindextrue
1148   \let\@gls\@glstr@p@text@
1149   \let\@Gls\@Gls@p@text@
1150   \let\@GLS\@GLS@p@text@
1151   \let\@glspl\@glstr@p@plural@
1152   \let\@Glspl\@Gls@p@plural@
1153   \let\@GLSpl\@GLS@p@plural@
1154   \let\@glstrshort\@glstr@p@short@
1155   \let\@Glsrshort\@Gls@p@short@
1156   \let\@GLSrshort\@GLS@p@short@
1157   \let\@glstrlong\@glstr@p@long@
1158   \let\@Glsrlong\@Gls@p@long@
1159   \let\@GLSrlong\@GLS@p@long@
1160   \let\@glstrshortpl\@glstr@p@shortpl@
1161   \let\@Glsrshortpl\@Gls@p@shortpl@
1162   \let\@GLSrshortpl\@GLS@p@shortpl@
1163   \let\@glstrlongpl\@glstr@p@longpl@
1164   \let\@Glsrlongpl\@Gls@p@longpl@
1165   \let\@GLSrlongpl\@GLS@p@longpl@
```

```

1166 \let\@acrshort\@glsxtrp@acrshort@
1167 \let\@Acrshort\@Glsxtrp@acrshort@
1168 \let\@ACRshort\@GLSxtrp@acrshort@
1169 \let\@acrshortpl\@glsxtrp@acrshortpl@
1170 \let\@Acrshortpl\@Glsxtrp@acrshortpl@
1171 \let\@ACRshortpl\@GLSxtrp@acrshortpl@
1172 \let\@acrlong\@glsxtrp@acrlong@
1173 \let\@Acrlong\@Glsxtrp@acrlong@
1174 \let\@ACRlong\@GLSxtrp@acrlong@
1175 \let\@acrlongpl\@glsxtrp@acrlongpl@
1176 \let\@Acrlongpl\@Glsxtrp@acrlongpl@
1177 \let\@ACRlongpl\@GLSxtrp@acrlongpl@
1178 }

```

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

@glsxtrp@text@

```
1179 \def\@glsxtrp@text@#1#2[#3]{\@glstext@{#1}{#2} [#3]}
```

@Glsxtrp@text@

```
1180 \def\@Glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2} [#3]}
```

@GLSxtrp@text@

```
1181 \def\@GLSxtrp@text@#1#2[#3]{\@GLStext@{#1}{#2} [#3]}
```

lsxtrp@plural@

```
1182 \def\@lsxtrp@plural@#1#2[#3]{\@glsplural@{#1}{#2} [#3]}
```

lsxtrp@plural@

```
1183 \def\@lsxtrp@plural@#1#2[#3]{\@Glsplural@{#1}{#2} [#3]}
```

LSxtrp@plural@

```
1184 \def\@LSxtrp@plural@#1#2[#3]{\@GLSplural@{#1}{#2} [#3]}
```

glsxtrp@short@

```

1185 \def\@glsxtrp@short@#1#2[#3]{%
1186 {%
1187   \glssetabbrvfmt{\glscategory{#2}}%
1188   \glsabbrvfont{\glsentryshort{#2}}#3%
1189 }%
1190 }

```

Glsxtrp@short@

```

1191 \def\@Glsxtrp@short@#1#2[#3]{%
1192 {%
1193   \glssetabbrvfmt{\glscategory{#2}}%
1194   \glsabbrvfont{\Glsentryshort{#2}}#3%
1195 }%
1196 }

```

GLSxtr@p@short@

```
1197 \def\@GLSxtr@p@short@#1#2[#3]{%
1198   {%
1199     \glsssetabbrvfmt{\glscategory{#2}}%
1200     \mfirstucMakeUppercase{\glssabbrvfont{\glssentryshort{#2}}#3}%
1201   }%
1202 }
```

sxtr@p@shortpl@

```
1203 \def\@glssxtr@p@shortpl@#1#2[#3]{%
1204   {%
1205     \glsssetabbrvfmt{\glscategory{#2}}%
1206     \glssabbrvfont{\glssentryshortpl{#2}}#3%
1207   }%
1208 }
```

Sxtr@p@shortpl@

```
1209 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
1210   {%
1211     \glsssetabbrvfmt{\glscategory{#2}}%
1212     \glssabbrvfont{\Glsentryshortpl{#2}}#3%
1213   }%
1214 }
```

Sxtr@p@shortpl@

```
1215 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
1216   {%
1217     \glsssetabbrvfmt{\glscategory{#2}}%
1218     \mfirstucMakeUppercase{\glssabbrvfont{\glssentryshortpl{#2}}#3}%
1219   }%
1220 }
```

@glssxtr@p@long@

```
1221 \def\@glssxtr@p@long@#1#2[#3]{\{\glssentrylong{#2}#3}\}
```

@Glsxtr@p@long@

```
1222 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}\}
```

@GLSxtr@p@long@

```
1223 \def\@GLSxtr@p@long@#1#2[#3]{%
1224   {\mfirstucMakeUppercase{\glssentrylong{#2}#3}}}
```

lsxtr@p@longpl@

```
1225 \def\@glssxtr@p@longpl@#1#2[#3]{\{\glssentrylongpl{#2}#3}\}
```

lsxtr@p@longpl@

```
1226 \def\@Glsxtr@p@longpl@#1#2[#3]{\{\Glsentrylongpl{#2}#3}\}
```

LSxtr@p@longpl@
1227 \def\@GLSxtr@p@longpl@#1#2[#3]{%
1228 {\mfirstucMakeUppercase{\glentrylongpl{#2}#3}}}

xtr@p@acrshort@
1229 \def\@glxtr@p@acrshort@#1#2[#3]{\acronymfont{\glentryshort{#2}#3}}

xtr@p@acrshort@
1230 \def\@Glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}#3}}

xtr@p@acrshort@
1231 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
1232 {\mfirstucMakeUppercase{\acronymfont{\glentryshort{#2}#3}}}

r@p@acrshortpl@
1233 \def\@glxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\glentryshortpl{#2}#3}}

r@p@acrshortpl@
1234 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}#3}}

r@p@acrshortpl@
1235 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
1236 {\mfirstucMakeUppercase{\acronymfont{\glentryshortpl{#2}#3}}}

sxtr@p@acrlong@
1237 \def\@glxtr@p@acrlong@#1#2[#3]{\glentrylong{#2}#3}}

sxtr@p@acrlong@
1238 \def\@Glsxtr@p@acrlong@#1#2[#3]{\Glsentrylong{#2}#3}}

Sxtr@p@acrlong@
1239 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
1240 {\mfirstucMakeUppercase{\glentrylong{#2}#3}}}

tr@p@acrlongpl@
1241 \def\@glxtr@p@acrlongpl@#1#2[#3]{\glentrylongpl{#2}#3}}

tr@p@acrlongpl@
1242 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\Glsentrylongpl{#2}#3}}

tr@p@acrlongpl@
1243 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
1244 {\mfirstucMakeUppercase{\glentrylongpl{#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.
1245 \renewcommand*{\@glsunset}[1]{%
1246   \@glsunset{#1}%
1247   \glsxtrpostunset{#1}%
1248 }%

glsxtrpostunset
1249 \newcommand*{\glsxtrpostunset}[1]{%

\@glslocalunset   Local unset.
1250 \renewcommand*{\@glslocalunset}[1]{%
1251   \@glslocalunset{#1}%
1252   \glsxtrpostlocalunset{#1}%
1253 }%

rpostlocalunset
1254 \newcommand*{\glsxtrpostlocalunset}[1]{%

\@glsreset   Global reset.
1255 \renewcommand*{\@glsreset}[1]{%
1256   \@glsreset{#1}%
1257   \glsxtrpostreset{#1}%
1258 }%

glsxtrpostreset
1259 \newcommand*{\glsxtrpostreset}[1]{%

\@glslocalreset   Local reset.
1260 \renewcommand*{\@glslocalreset}[1]{%
1261   \@glslocalreset{#1}%
1262   \glsxtrpostlocalreset{#1}%
1263 }%

rpostlocalreset
1264 \newcommand*{\glsxtrpostlocalreset}[1]{%

leEntryCounting   The first argument is the list of categories and the second argument is the value of the entrycount attribute.
1265 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```


Enable entry counting:

```
1266 \glsenableentrycount
```

Redefine \gls etc:

```
1267 \renewcommand*{\gls}{\cglsl}%
1268 \renewcommand*{\Gls}{\cGls}%
1269 \renewcommand*{\glspl}{\cglspl}%
1270 \renewcommand*{\Glspl}{\cGlspl}%
1271 \renewcommand*{\GLS}{\cGLS}%
1272 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
1273 \@glsxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
1274 \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
1275 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
1276   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
1277     can't be used with \string\GlsXtrEnableEntryCounting}%
1278   {Use one or other but not both commands}}%
1279 }
```

ycountunsetattr

```
1280 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
1281   \@for\@glsxtr@cat:=#1\do
1282   {%
1283     \ifdefempty{\@glsxtr@cat}{}%
1284     {%
1285       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
1286     }%
1287   }%
1288 }
```

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

nableentrycount

```
1289 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
1290 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
```

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

```
1291 \renewcommand*{\gls@defdocnewglossaryentry}{%
1292   \renewcommand*{\newglossaryentry}[2]{%
1293     \PackageError{glossaries}{\string\newglossaryentry\space
1294       may only be used in the preamble when entry counting has
1295       been activated}{If you use \string\glsenableentrycount\space
1296       you must place all entry definitions in the preamble not in
1297       the document environment}%
1298   }%
1299 }
```

New commands to access new fields:

```

1300 \newcommand*{\glsentrycurrcount}[1]{%
1301   \ifcsundef{glo@glsdetoklabel{##1}@currcount}%
1302   {0}{\@gls@entry@field{##1}{currcount}}%
1303 }%
1304 \newcommand*{\glsentryprevcount}[1]{%
1305   \ifcsundef{glo@glsdetoklabel{##1}@prevcount}%
1306   {0}{\@gls@entry@field{##1}{prevcount}}%
1307 }%

```

Adjust post unset and reset:

```

1308 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
1309 \renewcommand*{\glsxtrpostunset}[1]{%
1310   \@glsxtr@entrycount@org@unset{##1}%
1311   \@gls@increment@currcount{##1}%
1312 }%
1313 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
1314 \renewcommand*{\glsxtrpostlocalunset}[1]{%
1315   \@glsxtr@entrycount@org@localunset{##1}%
1316   \@gls@local@increment@currcount{##1}%
1317 }%
1318 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
1319 \renewcommand*{\glsxtrpostreset}[1]{%
1320   \@glsxtr@entrycount@org@reset{##1}%
1321   \csgdef{glo@glsdetoklabel{##1}@currcount}{0}%
1322 }%
1323 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
1324 \renewcommand*{\glsxtrpostlocalreset}[1]{%
1325   \@glsxtr@entrycount@org@localreset{##1}%
1326   \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
1327 }%

```

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

```

1328 \let\@cgl@s\@cgl@s
1329 \let\@cgl spl\@cgl spl
1330 \let\@cGLS\@cGLS
1331 \let\@cGL spl\@cGL spl
1332 \let\@cGLS\@cGLS
1333 \let\@cGLSpl\@cGLSpl

```

The rest is as the original definition.

```

1334 \AtEndDocument{\@gls@write@entrycounts}%
1335 \renewcommand*{\@gls@entry@count}[2]{%
1336   \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
1337 }%
1338 \let\glsenableentrycount\relax
1339 \renewcommand*{\glsenableentryunitcount}{%
1340   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space
1341     can't be used with \string\glsenableentrycount}%

```

```

1342     {Use one or other but not both commands}%
1343   }%
1344 }

```

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

```

1345 \renewcommand*{\@gls@write@entrycounts}{%
1346   \immediate\write\@auxout
1347     {\string\providecommand*{\string\@gls@entry@count}[2]{}}%
1348   \count@=0\relax
1349   \forallglsentries{\@glsentry}{%
1350     \glshasattribute{\@glsentry}{entrycount}%
1351     {%
1352       \ifglsused{\@glsentry}%
1353       {%
1354         \immediate\write\@auxout
1355           {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
1356       }%
1357     }%
1358     \advance\count@ by \@ne
1359   }%
1360 }%
1361 }%
1362 \ifnum\count@=0
1363   \GlossariesExtraWarningNoLine{Entry counting has been enabled
1364     \MessageBreak with \string\glsenableentrycount\space but the
1365     \MessageBreak attribute 'entrycount' hasn't
1366     \MessageBreak been assigned to any of the defined
1367     \MessageBreak entries}%
1368 \fi
1369 }

```

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

```

1370 \newcommand*{\glxtrifcounttrigger}[3]{%
1371   \glshasattribute{#1}{entrycount}%
1372   {%
1373     \ifnum\glsentryprevcount{#1}>\glsgetattribute{#1}{entrycount}\relax
1374     #3%
1375   \else
1376     #2%
1377   \fi
1378 }%
1379 {#3}%
1380 }

```

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

\@@cgl's@

```

1381 \def\@@cgl's@#1#2[#3]{%
1382   \gl'sxtrifcounttrigger{#2}%
1383   {%
1384     \cgl'sformat{#2}{#3}%
1385     \gl'sunset{#2}%
1386   }%
1387   {%
1388     \@gl's@{#1}{#2}[#3]%
1389   }%
1390 }%
```

\@@cgl'spl@

```

1391 \def\@@cgl'spl@#1#2[#3]{%
1392   \gl'sxtrifcounttrigger{#2}%
1393   {%
1394     \cgl'splformat{#2}{#3}%
1395     \gl'sunset{#2}%
1396   }%
1397   {%
1398     \@gl'spl@{#1}{#2}[#3]%
1399   }%
1400 }%
```

\@@cGl's@

```

1401 \def\@@cGl's@#1#2[#3]{%
1402   \gl'sxtrifcounttrigger{#2}%
1403   {%
1404     \cGl'sformat{#2}{#3}%
1405     \gl'sunset{#2}%
1406   }%
1407   {%
1408     \@Gl's@{#1}{#2}[#3]%
1409   }%
1410 }%
```

\@@cGl'spl@

```

1411 \def\@@cGl'spl@#1#2[#3]{%
1412   \gl'sxtrifcounttrigger{#2}%
1413   {%
1414     \cGl'splformat{#2}{#3}%
1415     \gl'sunset{#2}%
1416   }%
1417   {%
1418     \@Gl'spl@{#1}{#2}[#3]%
1419   }%
1420 }%
```

```

\@cGLS@
1421 \def\@cGLS@#1#2[#3]{%
1422   \glxtrifcounttrigger{#2}%
1423   {%
1424     \cGLSformat{#2}{#3}%
1425     \glset{#2}%
1426   }%
1427   {%
1428     \@GLS@{#1}{#2}[#3]%
1429   }%
1430 }%

\@cGLSpl@
1431 \def\@cGLSpl@#1#2[#3]{%
1432   \glxtrifcounttrigger{#2}%
1433   {%
1434     \cGLSplformat{#2}{#3}%
1435     \glset{#2}%
1436   }%
1437   {%
1438     \@GLSpl@{#1}{#2}[#3]%
1439   }%
1440 }%
1441 %
1442 % Remove default warnings from \cs{cgl} etc so that it can be used
1443 % interchangeable with \cs{gl} etc.
1444 %\begin{macro}\@cgl@
1445 %   \begin{macrocode}
1446 \def\@cgl@#1#2[#3]{\@gl@{#1}{#2}[#3]}

\@cGl@
1447 \def\@cGl@#1#2[#3]{\@gl@{#1}{#2}[#3]}

\@cglspl@
1448 \def\@cglspl@#1#2[#3]{\@glspl@{#1}{#2}[#3]}

\@cGlspl@
1449 \def\@cGlspl@#1#2[#3]{\@glspl@{#1}{#2}[#3]}

    Add all upper case versions not provided by glossaries.

\cGLS
1450 \newrobustcmd*{\cGLS}{\@glshyp@opt\@cGLS}

\@cGLS   Defined the un-starred form. Need to determine if there is a final optional argument
1451 \newcommand*{\@cGLS}[2][{}]{%
1452   \new@ifnextchar[\@cGLS@{#1}{#2}]{\@cGLS@{#1}{#2}[{}]}%
1453 }

```

```

\@cGLS@
1454 \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.
1455 \newcommand*{\cGLSformat}[2]{%
1456   \expandafter\mfirstucMakeUppercase\expandafter{\cglformat{#1}{#2}}%
1457 }

\cGLSpl
1458 \newrobustcmd*{\cGLSpl}{\@gls@hyp@opt\@cGLSpl}

\@cGLSpl  Defined the un-starred form. Need to determine if there is a final optional argument
1459 \newcommand*{\@cGLSpl}[2][{}]{%
1460   \new@ifnextchar[\@cGLSpl@{#1}{#2}]{\@cGLSpl@{#1}{#2}[{}]}%
1461 }

\@cGLSpl@
1462 \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.
1463 \newcommand*{\cGLSplformat}[2]{%
1464   \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
1465 }

      Modify the trigger formats to check for the regular attribute.

\cglformat
1466 \renewcommand*{\cglformat}[2]{%
1467   \glsifregular{#1}
1468   {\glsentryfirst{#1}}%
1469   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
1470 }

\cGlsformat
1471 \renewcommand*{\cGlsformat}[2]{%
1472   \glsifregular{#1}
1473   {\Glsentryfirst{#1}}%
1474   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
1475 }

\cglsplformat
1476 \renewcommand*{\cglsplformat}[2]{%
1477   \glsifregular{#1}
1478   {\glsentryfirstplural{#1}}%
1479   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
1480 }

```

\cGlsplformat

```
1481 \renewcommand*{\cGlsplformat}[2]{%
1482   \glsifregular{#1}
1483   {\Glsentryfirstplural{#1}}%
1484   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2}%
1485 }
```

New code similar to above for unit counting.

defunitcounters

```
1486 \newcommand*{\@@newglossaryentry@defunitcounters}{%
1487   \edef\@glo@countunit{\csuse{\glstr@categoryattr@@\@glo@category @unitcount}}%
1488   \ifvoid\@glo@countunit
1489   {}%
1490   {%
1491     \@glstr@ifunitcounter{\@glo@countunit}%
1492     {}%
1493     {\expandafter\@glstr@addunitcounter\expandafter{\@glo@countunit}}%
1494   }%
1495 }
```

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

```
1496 \newcommand*{\@glstr@unitcountlist}{}%
```

@addunitcounter

```
1497 \newcommand*{\@glstr@addunitcounter}[1]{%
1498   \listadd{\@glstr@unitcountlist}{#1}%
1499   \ifcsundef{glstr@theunit@#1}
1500   {%
1501     \ifcsdef{theH#1}%
1502     {\csdef{glstr@theunit@#1}{\csuse{theH#1}}}%
1503     {\csdef{glstr@theunit@#1}{\csuse{the#1}}}%
1504   }%
1505   {}%
1506 }
```

r@ifunitcounter

```
1507 \newcommand*{\@glstr@ifunitcounter}[3]{%
1508   \xifinlist{#1}{\@glstr@unitcountlist}{#2}{#3}%
1509 }
```

urrentunitcount

```
1510 \newcommand*{\@glstr@currentunitcount}[1]{%
1511   glo@\glstetoklabel{#1}@currunit@\glstetattribute{#1}{unitcount}.%
1512   \csuse{glstr@theunit@\glstetattribute{#1}{unitcount}}%
1513 }
```

previousunitcount

```
1514 \newcommand*\@glxtr@previousunitcount[1]{%
1515   glo@glstdetoklabel{#1}@prevunit@glsggetattribute{#1}{unitcount}.%
1516   \csuse{glxtr@theunit@glsggetattribute{#1}{unitcount}}%
1517 }
```

t@currunitcount

```
1518 \newcommand*\@glxtr@increment@currunitcount[1]{%
1519   \glshasattribute{#1}{unitcount}%
1520   {%
1521     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
1522     \ifcsundef{\@glxtr@csname}%
1523     {%
1524       \csgdef{\@glxtr@csname}{1}%
1525       \listcsxadd
1526       {glo@glstdetoklabel{#1}@unitlist}%
1527       {\glsggetattribute{#1}{unitcount}.%
1528        \csuse{glxtr@theunit@glsggetattribute{#1}{unitcount}}}%
1529     }%
1530   }%
1531   {%
1532     \csxdef{\@glxtr@csname}%
1533     {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
1534   }%
1535 }%
1536 {}%
1537 }
```

t@currunitcount

```
1538 \newcommand*\@glxtr@local@increment@currunitcount[1]{%
1539   \glshasattribute{#1}{unitcount}%
1540   {%
1541     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
1542     \ifcsundef{\@glxtr@csname}%
1543     {%
1544       \csdef{\@glxtr@csname}{1}%
1545       \listcseadd
1546       {glo@glstdetoklabel{#1}@unitlist}%
1547       {\glsggetattribute{#1}{unitcount}.%
1548        \csuse{glxtr@theunit@glsggetattribute{#1}{unitcount}}}%
1549     }%
1550   }%
1551   {%
1552     \csedef{\@glxtr@csname}%
1553     {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
1554   }%
1555 }%
1556 {}%
1557 }
```


r@currunitcount

```
1558 \newcommand*{\@glxstr@currunitcount}[2]{%
1559   \ifcsundef
1560   {glo@\glstdetoklabel{#1}@currunit@#2}%
1561   {0}%
1562   {\csuse{glo@\glstdetoklabel{#1}@currunit@#2}}%
1563 }%
```

r@prevunitcount

```
1564 \newcommand*{\@glxstr@prevunitcount}[2]{%
1565   \ifcsundef
1566   {glo@\glstdetoklabel{#1}@prevunit@#2}%
1567   {0}%
1568   {\csuse{glo@\glstdetoklabel{#1}@prevunit@#2}}%
1569 }%
```

eentryunitcount

```
1570 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
1571   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
1572   \renewcommand*{\glstocnewglossaryentry}{%
1573     \renewcommand*\newglossaryentry[2]{%
1574       \PackageError{glossaries}{\string\newglossaryentry\space
1575       may only be used in the preamble when entry counting has
1576       been activated}{If you use \string\glsenableentryunitcount\space
1577       you must place all entry definitions in the preamble not in
1578       the document environment}%
1579     }%
1580   }%

  New commands to access new fields:
1581   \newcommand*{\glstoccurrunitcount}[1]{%
1582     \@glxstr@currunitcount{##1}{\glstocattribute{##1}{unitcount}}.%
1583     \csuse{glstoccurrunit@theunit@\glstocattribute{##1}{unitcount}}}%
1584   }%
1585   \newcommand*{\glstocprevunitcount}[1]{%
1586     \@glxstr@prevunitcount{##1}{\glstocattribute{##1}{unitcount}}.%
1587     \csuse{glstocprevunit@theunit@\glstocattribute{##1}{unitcount}}}%
1588   }%

  Access total count:
1589   \newcommand*{\glstocprevtotalcount}[1]{%
1590     \ifcsundef{glo@\glstdetoklabel{##1}@prevunittotal}%
1591     {0}%
1592     {%
1593       \number\csuse{glo@\glstdetoklabel{##1}@prevunittotal}
1594     }%
1595   }%
```

Access max value:

```

1596 \newcommand*{\glstentryprevmaxcount}[1]{%
1597   \ifcsundef{glo@\glstetoklabel{##1}@prevunitmax}%
1598     {}%
1599     {%
1600       \number\csuse{glo@\glstetoklabel{##1}@prevunitmax}
1601     }%
1602   }%

```

Adjust post unset and reset:

```

1603 \let\@glstxtr@entryunitcount@org@unset\glstxtrpostunset
1604 \renewcommand*{\glstxtrpostunset}[1]{%
1605   \@glstxtr@entryunitcount@org@unset{##1}%
1606   \@glst@increment@currunitcount{##1}%
1607 }%
1608 \let\@glstxtr@entryunitcount@org@localunset\glstxtrpostlocalunset
1609 \renewcommand*{\glstxtrpostlocalunset}[1]{%
1610   \@glstxtr@entryunitcount@org@localunset{##1}%
1611   \@glst@local@increment@currunitcount{##1}%
1612 }%
1613 \let\@glstxtr@entryunitcount@org@reset\glstxtrpostreset
1614 \renewcommand*{\glstxtrpostreset}[1]{%
1615   \glshasattribute{##1}{unitcount}%
1616   {%
1617     \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
1618     \ifcsundef{\@glstxtr@csname}%
1619       {}%
1620       {\csgdef{\@glstxtr@csname}{0}}%
1621     }%
1622   {}%
1623 }%
1624 \let\@glstxtr@entryunitcount@org@localreset\glstxtrpostlocalreset
1625 \renewcommand*{\glstxtrpostlocalreset}[1]{%
1626   \@glstxtr@entryunitcount@org@localreset{##1}%
1627   \glshasattribute{##1}{unitcount}%
1628   {%
1629     \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
1630     \ifcsundef{\@glstxtr@csname}%
1631       {}%
1632       {\csdef{\@glstxtr@csname}{0}}%
1633     }%
1634   {}%
1635 }%

```

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

```

1636 \let\@cglst@\@cglst@
1637 \let\@cglstpl@\@cglstpl@
1638 \let\@cGLS@\@cGLS@
1639 \let\@cGLstpl@\@cGLstpl@

```

```

1640 \let\@cGLS@ \@cGLS@
1641 \let\@cGLSpl@ \@cGLSpl@

Write information to the aux file.

1642 \AtEndDocument{\@gls@write@entryunitcounts}%
1643 \renewcommand*{\@gls@entry@unitcount}[3]{%
1644   \csgdef{glo@\glsdetoklabel{##1}@prevunit@##3}{##2}%
1645   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
1646   {\csgdef{glo@\glsdetoklabel{##1}@prevunittotal}{##2}}%
1647   {%
1648     \csxdef{glo@\glsdetoklabel{##1}@prevunittotal}{
1649       \number\numexpr\csuse{glo@\glsdetoklabel{##1}@prevunittotal}+##2}%
1650     }%
1651     \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
1652     {\csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}}%
1653     {%
1654       \ifnum\csuse{glo@\glsdetoklabel{##1}@prevunitmax}<##2
1655       \csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}%
1656       \fi
1657     }%
1658   }%
1659 \let\glsenableentryunitcount\relax
1660 \renewcommand*{\glsenableentrycount}{%
1661   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
1662     can't be used with \string\glsenableentryunitcount}%
1663   {Use one or other but not both commands}%
1664 }%
1665 }
1666 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

1667 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

1668 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
1669   \immediate\write\@auxout
1670   {\string\@gls@entry@unitcount
1671     {\@glsentry}%
1672     {\@glsxtr@currunitcount{\@glsentry}{#1}%
1673     }%
1674     {#1}}%
1675 }

```

entryunitcounts

```

1676 \newcommand*{\@gls@write@entryunitcounts}{%
1677   \immediate\write\@auxout
1678   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
1679   \count@=0\relax
1680   \forallglsentries{\@glsentry}{%

```

```

1681 \glsattribute{\@glsentry}{unitcount}%
1682 {%
1683 \ifglsused{\@glsentry}%
1684 {%
1685 \forlistcsloop
1686 {\@gls@write@entryunitcounts@do}%
1687 {glo@\glsdetoklabel{\@glsentry}@unitlist}%
1688 }%
1689 {}%
1690 \advance\count@ by \@ne
1691 }%
1692 {}%
1693 }%
1694 \ifnum\count@=0
1695 \GlossariesExtraWarningNoLine{Entry counting has been enabled
1696 \MessageBreak with \string\glsenableentryunitcount\space but the
1697 \MessageBreak attribute ‘unitcount’ hasn’t
1698 \MessageBreak been assigned to any of the defined
1699 \MessageBreak entries}%
1700 \fi
1701 }

```

`\glsenableentryunitcount` 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.

```

1702 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

1703 \glsenableentryunitcount

```

Redefine `\gls` etc:

```

1704 \renewcommand*{\gls}{\cgl}%
1705 \renewcommand*{\Gls}{\cGls}%
1706 \renewcommand*{\glspl}{\cglspl}%
1707 \renewcommand*{\Glspl}{\cGlspl}%
1708 \renewcommand*{\GLS}{\cGLS}%
1709 \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

1710 \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

1711 \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
1712 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
1713 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
1714 can’t be used with \string\GlsXtrEnableEntryUnitCounting}%
1715 {Use one or other but not both commands}}%
1716 }

```

`\glsxtr@setentryunitcountunsetattr`

```

1717 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
1718 \@for\@glsxtr@cat:=#1\do

```

```

1719 {%
1720   \ifdefempty{\@glstr@cat}{}%
1721   {%
1722     \glsssetcategoryattribute{\@glstr@cat}{entrycount}{#2}%
1723     \glsssetcategoryattribute{\@glstr@cat}{unitcount}{#3}%
1724   }%
1725 }%
1726 }

```

1.3.6 Acronym Modifications

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

`\GenericNewAcronym`

```

1727 \renewcommand*{\SetGenericNewAcronym}{%
1728   \let\@Gls@entryname\@Gls@acrenryname
1729   \renewcommand{\newacronym}[4][]{%
1730     \ifdefempty{\@glssacronymlists}%
1731     {%
1732       \def\@glo@type{\acronymtype}%
1733       \setkeys{glossentry}{##1}%
1734       \DeclareAcronymList{\@glo@type}%
1735     }%
1736   }%
1737   \glsskeylisttok{##1}%
1738   \glsslabeltok{##2}%
1739   \glssshorttok{##3}%
1740   \glsslongtok{##4}%
1741   \newacronymhook
1742   \protected@edef\@do@newglossaryentry{%
1743     \noexpand\newglossaryentry{\the\glsslabeltok}%
1744     {%
1745       type=\acronymtype,%
1746       name={\expandonce{\acronymentry{##2}}},%
1747       sort={\acronymssort{\the\glssshorttok}{\the\glsslongtok}},%
1748       text={\the\glssshorttok},%
1749       short={\the\glssshorttok},%
1750       shortplural={\the\glssshorttok\noexpand\acrpluralsuffix},%
1751       long={\the\glsslongtok},%
1752       longplural={\the\glsslongtok\noexpand\acrpluralsuffix},%
1753       category=acronym,%
1754       \GenericAcronymFields,%
1755       \the\glsskeylisttok

```

```

1756     }%
1757 }%
1758 \do@newglossaryentry
1759 }%
1760 \renewcommand*{\acrfullfmt}[3]{%
1761   \glslink{##1}{##2}{\genacrfullformat{##2}{##3}}}%
1762 \renewcommand*{\Acrfullfmt}[3]{%
1763   \glslink{##1}{##2}{\Genacrfullformat{##2}{##3}}}%
1764 \renewcommand*{\ACRfullfmt}[3]{%
1765   \glslink{##1}{##2}{%
1766     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
1767 \renewcommand*{\acrfullplfmt}[3]{%
1768   \glslink{##1}{##2}{\genplacrfullformat{##2}{##3}}}%
1769 \renewcommand*{\Acrfullplfmt}[3]{%
1770   \glslink{##1}{##2}{\Genplacrfullformat{##2}{##3}}}%
1771 \renewcommand*{\ACRfullplfmt}[3]{%
1772   \glslink{##1}{##2}{%
1773     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
1774 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}}%
1775 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
1776 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}}%
1777 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
1778 }

```

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:

```

1779 \let\@glsxtr@org@setacronymstyle\setacronymstyle
1780 \let\@glsxtr@org@newacronymstyle\newacronymstyle

```

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

```

1781 \newcommand*{\MakeAcronymsAbbreviations}{%
1782   \renewcommand*{\newacronym}[4][ ]{%
1783     \newabbreviation[type=\acronymtype,category=acronym,##1]{##2}{##3}{##4}%
1784   }%
1785   \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
1786   \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%
1787   \renewcommand*{\setacronymstyle}[1]{%
1788     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}
1789     unavailable.
1790     Use \string\setabbreviationstyle\space instead.
1791     The original acronym interface can be restored with
1792     \string\RestoreAcronyms}{}%
1793 }%
1794 \renewcommand*{\newacronymstyle}[1]{%
1795   \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
1796   available unless you restore the original acronym interface with

```

```

1797 \string\RestoreAcronyms}%
1798 \@glsxtr@org@newacronymstyle{##1}%
1799 }%
1800 }

```

Switch acronyms to abbreviations:

```
1801 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```

1802 \newcommand*{\RestoreAcronyms}{%
1803 \SetGenericNewAcronym
1804 \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
1805 \renewcommand{\acronymfont}[1]{##1}%
1806 \let\setacronymstyle\@glsxtr@org@setacronymstyle
1807 \let\newacronymstyle\@glsxtr@org@newacronymstyle
1808 \let\@gls@link@checkfirsthyper\@glsxtr@org@checkfirsthyper
1809 \glssetcategoryattribute{acronym}{regular}{false}%
1810 \setacronymstyle{long-short}%
1811 }

```

\glsacspace Allow the user to customise the maximum value.

```

1812 \renewcommand*{\glsacspace}[1]{%
1813 \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
1814 \ifdim\dimen@<\glsacspacemax~\else\space\fi
1815 }

```

\glsacspacemax Value used in the above.

```
1816 \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

```
1817 \newcommand*{\@glsxtr@reg@glosslist}{}
```

Save the original definition of **\makeglossaries**:

```
1818 \let\@glsxtr@org@makeglossaries\makeglossaries
```

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

\makeglossaries

```

1819 \renewcommand*{\makeglossaries}[1][]{%
1820   \ifblank{#1}%
1821   {\@glstr@org@makeglossaries}%
1822   {%
1823     \edef\@glstr@reg@glosslist{#1}%
1824     \ifundef{\glswrite}{\newwrite\glswrite}{}%
1825     \protected@write\@auxout{}{\string\providecommand
1826       \string\@glstr@order[1]}%
1827     \protected@write\@auxout{}{\string\providecommand
1828       \string\@istfilename[1]}%
1829     \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
1830     \protected@write\@auxout{}{\string\@glstr@order{\glstr@order}}
1831     \write\@auxout{\string\providecommand\string\@glstr@reference[3]}%

```

Iterate through each supplied glossary type and activate it.

```

1832   \@for\@glo@type:=#1\do{%
1833     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
1834   }%

```

New glossaries must be created before \makeglossaries:

```

1835   \renewcommand*{\newglossary}[4][]{%
1836     \PackageError{glossaries}{New glossaries
1837       must be created before \string\makeglossaries}{You need
1838       to move \string\makeglossaries\space after all your
1839       \string\newglossary\space commands}}%

```

Any subsequent instances of this command should have no effect

```

1840   \let\@makeglossary\relax
1841   \let\makeglossary\relax
1842   \let\makeglossaries\relax

```

Disable all commands that have no effect after \makeglossaries

```

1843   \@disable@onlypremakeg

```

Allow see key:

```

1844   \let\glstr@checkseeallowed\relax

```

Suppress warning about no \makeglossaries

```

1845   \let\warn@nomakeglossaries\relax
1846   \def\warn@noprintglossary{%
1847     \GlossariesWarningNoLine{No \string\printglossary\space
1848       or \string\printglossaries\space
1849       found.^^J(Remove \string\makeglossaries\space if you don't
1850 want
1851       any glossaries.)^^JThis document will not have a glossary}%
1852   }%

```

Adjust display number list to check for type:

```

1853   \renewcommand*{\glstrdisplaynumberlist}[1]{%
1854     \expandafter\DTLifinlist\expandafter{##1}{\@glstr@reg@glosslist}%
1855     {\@glstr@idx@displaynumberlist{##1}}%

```



```

1856   {\@glxtr@noidx@displaynumberlist{##1}}}%
1857 }%

```

Adjust entry list:

```

1858 \renewcommand*{\glstrynumberlist}[1]{%
1859   \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
1860   {\@glxtr@idx@entrynumberlist{##1}}}%
1861   {\@glxtr@noidx@entrynumberlist{##1}}}%
1862 }%

```

Adjust number list loop

```

1863 \renewcommand*{\glsnnumberlistloop}[2]{%
1864   \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
1865   {%
1866     \PackageError{glossaries-extra}{\string\glsnnumberlistloop\space
1867       not available for glossary ‘##1’}{}%
1868   }%
1869   {\@glxtr@noidx@numberlistloop{##1}{##2}}}%
1870 }%

```

Only sanitize sort for normal indexing glossaries.

```

1871 \renewcommand*{\glspstandardsort}[3]{%
1872   \expandafter\DTLifinlist\expandafter{##2}{\@glxtr@reg@glosslist}%
1873   {%
1874     \glsdosanitizesort
1875   }%
1876   {%
1877     \ifglssanitizesort
1878       \@gl@noidx@sanitizesort
1879     \else
1880       \@gl@noidx@nosanitizesort
1881     \fi
1882   }%
1883 }%

```

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

```

1884 \renewcommand*{\new@glossaryentry}[2]{%
1885   \PackageError{glossaries-extra}{Glossary entries must be defined
1886     in the preamble\MessageBreak when you use the optional argument
1887     of \string\makeglossaries}{Either move your definitions to the
1888     preamble or don't use the optional argument of
1889     \string\makeglossaries}%
1890 }%

```

Only activate sort key for glossaries that aren't listed in #1 (glossary label is stored in \@glo@type but this defaults to \gldefaulttype so some expansion is required).

```

1891 \renewcommand*{\@printgloss@setsort}{%
1892   \renewcommand*{\@glo@assign@sortkey}{%
1893     \edef\@glo@type{\@glo@type}%
1894     \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%

```

```

1895     {%
1896         \@glo@no@assign@sortkey
1897     }%
1898     {%
1899         \@glo@assign@sortkey
1900     }%
1901 }%
1902 \def\@glo@sorttype{\@glo@default@sorttype}%
1903 }%

```

Check automake setting:

```

1904 \ifglsautomake
1905 \renewcommand*{\@gls@doautomake}{%
1906     \@for\@gls@type:=\@glsxtr@reg@glosslist\do{%
1907         \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
1908     }%
1909 }%
1910 \fi
1911 }%
1912 }

```

Display number list for the regular version:

splaynumberlist

```

1913 \let\@glsxtr@idx@displaynumberlist\glsdisplaynumberlist

```

Display number list for the “noidx” version:

splaynumberlist

```

1914 \newcommand*{\@glsxtr@noidx@displaynumberlist}[1]{%
1915     \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
1916     \ifdef\@gls@loclist
1917     {%
1918         \def\@gls@noidxloclist@sep{%
1919             \def\@gls@noidxloclist@sep{%
1920                 \def\@gls@noidxloclist@sep{%
1921                     \glsnumlistsep
1922                 }%
1923                 \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
1924             }%
1925         }%
1926         \def\@gls@noidxloclist@finalsep{}%
1927         \def\@gls@noidxloclist@prev{}%
1928         \forlistloop{\@glsnoidxdisplayloclisthandler}{\@gls@loclist}%
1929         \@gls@noidxloclist@finalsep
1930         \@gls@noidxloclist@prev
1931     }%
1932     {%
1933         ??\glsdoifexists{#1}%
1934         {%
1935             \GlossariesWarning{Missing location list for ‘#1’. Either

```

```

1936         a rerun is required or you haven't referenced the entry.}%
1937     }%
1938 }%
1939 }%
1940

```

And for the number list loop:

@numberlistloop

```

1941 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
1942   \letcs{\@gls@loclist}{glo@glsetoklabel{#1}@loclist}%
1943   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
1944   \let\@gls@org@glssееformat\glssееformat
1945   \let\glsnoidxdisplayloc#2\relax
1946   \let\glssееformat#3\relax
1947   \ifdef\@gls@loclist
1948   {%
1949     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
1950   }%
1951   {%
1952     ??\glsoifexists{#1}%
1953     {%
1954       \GlossariesWarning{Missing location list for '#1'. Either
1955         a rerun is required or you haven't referenced the entry.}%
1956     }%
1957   }%
1958   \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
1959   \let\glssееformat\@gls@org@glssееformat
1960 }%

```

Same for entry number list.

entrynumberlist

```

1961 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
1962   \letcs{\@gls@loclist}{glo@glsetoklabel{#1}@loclist}%
1963   \ifdef\@gls@loclist
1964   {%
1965     \glsnoidxloclist{\@gls@loclist}%
1966   }%
1967   {%
1968     ??\glsoifexists{#1}%
1969     {%
1970       \GlossariesWarning{Missing location list for '#1'. Either
1971         a rerun is required or you haven't referenced the entry.}%
1972     }%
1973   }%
1974 }%

```

entrynumberlist

```

1975 \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

```

1976 \renewcommand{\@print@glossary}{%
1977   \makeatletter
1978   \@input@{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
1979   \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
1980   {}%
1981   {\glstrNoGlossaryWarning{\@glo@type}}%
1982   \ifglxindy
1983     \ifcsundef{@xdy@\@glo@type @language}%
1984     {%
1985       \edef\@do@auxoutstuff{%
1986         \noexpand\AtEndDocument{%
1987           \noexpand\immediate\noexpand\write\@auxout{%
1988             \string\providecommand\string\@xdylanguage[2]{}%
1989             \noexpand\immediate\noexpand\write\@auxout{%
1990               \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
1991             }%
1992           }%
1993         }%
1994       }%
1995       \edef\@do@auxoutstuff{%
1996         \noexpand\AtEndDocument{%
1997           \noexpand\immediate\noexpand\write\@auxout{%
1998             \string\providecommand\string\@xdylanguage[2]{}%
1999             \noexpand\immediate\noexpand\write\@auxout{%
2000               \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
2001                 @language\endcsname}}%
2002             }%
2003           }%
2004         }%
2005       \@do@auxoutstuff
2006       \edef\@do@auxoutstuff{%
2007         \noexpand\AtEndDocument{%
2008           \noexpand\immediate\noexpand\write\@auxout{%
2009             \string\providecommand\string\@gls@codepage[2]{}%
2010             \noexpand\immediate\noexpand\write\@auxout{%
2011               \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
2012             }%
2013           }%
2014         \@do@auxoutstuff
2015       \fi
2016       \renewcommand*{\@warn@nomakeglossaries}{%
2017         \GlossariesWarningNoLine{\string\makeglossaries\space
2018           hasn't been used,^^Jthe glossaries will not be updated}%
2019       }%
2020 }

```

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

`oGlsWarningHead` Header message.

```
2021 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
2022   This document is incomplete. The external file associated with
2023   the glossary ‘#1’ (which should be called \texttt{#2})
2024   hasn’t been created.%
2025 }
```

`arningEmptyStart` No entries have been added to the glossary.

```
2026 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
2027   This has probably happened because there are no entries defined
2028   in this glossary.%
2029 }
```

`arningEmptyMain` The default “main” glossary is empty.

```
2030 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
2031   If you don’t want this glossary,
2032   add \texttt{nomain} to your package option list when you load
2033   \texttt{glossaries-extra.sty}. For example:%
2034 }
```

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

```
2035 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
2036   Did you forget to use \texttt{type=#1} when you defined your
2037   entries? If you tried to load entries into this glossary with
2038   \texttt{\string\loadglsentries} did you remember to use
2039   \texttt{[#1]} as the optional argument? If you did, check that
2040   the definitions in the file you loaded all had the type set
2041   to \texttt{\string\glsdefaulttype}.%
2042 }
```

`arningCheckFile` Advisory message to check the file contents.

```
2043 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
2044   Check the contents of the file \texttt{#1}. If
2045   it’s empty, that means you haven’t indexed any of your entries in this
2046   glossary (using commands like \texttt{\string\gls} or
2047   \texttt{\string\glsadd}) so this list can’t be generated.
2048   If the file isn’t empty, the document build process hasn’t been
2049   completed.%
2050 }
```

`WarningAutoMake` Message when automake option has been used.

```
2051 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
2052   You may need to rerun \LaTeX. If you already have, it may be that
2053   \TeX’s shell escape doesn’t allow you to run
2054   \ifglxindy xindy\else makeindex\fi. Check the
2055   transcript file \texttt{\jobname.log}. If the shell escape is
```

```

2056 disabled, try one of the following:
2057
2058 \begin{itemize}
2059   \item Run the external (Lua) application:
2060
2061       \texttt{makeglossaries-lite \string"\jobname\string"}
2062
2063   \item Run the external (Perl) application:
2064
2065       \texttt{makeglossaries \string"\jobname\string"}
2066 \end{itemize}
2067
2068 Then rerun \LaTeX\ on this document.
2069 \GlossariesExtraWarning{Rerun required to build the
2070 glossary ‘#1’ or check TeX’s shell escape allows
2071 you to run \ifglxindy xindy\else makeindex\fi}%
2072 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

2073 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
2074   You need to either replace \texttt{\string\makenoidxglossaries}
2075   with \texttt{\string\makeglossaries} or replace
2076   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
2077   \texttt{\string\printnoidxglossary}
2078   (or \texttt{\string\printnoidxglossaries}) and then rebuild
2079   this document.%
2080 }

```

arningBuildInfo Build advice.

```

2081 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
2082   Try one of the following:
2083   \begin{itemize}
2084     \item Add \texttt{automake} to your package option list when you load
2085           \texttt{glossaries-extra.sty}. For example:
2086
2087           \texttt{\string\usepackage[automake]%
2088                 \glsgopenbrace glossaries-extra\glsgclosebrace}
2089
2090     \item Run the external (Lua) application:
2091
2092           \texttt{makeglossaries-lite \string"\jobname\string"}
2093
2094     \item Run the external (Perl) application:
2095
2096           \texttt{makeglossaries \string"\jobname\string"}
2097   \end{itemize}
2098
2099   Then rerun \LaTeX\ on this document.%
2100 }

```

oGlsWarningTail Final paragraph.

```
2101 \newcommand{\GlsXtrNoGlsWarningTail}{%
2102   This message will be removed once the problem has been fixed.%
2103 }
```

GlsWarningNoOut No out file created. Build advice.

```
2104 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
2105   The file \texttt{#1} doesn't exist. This most likely means you haven't used
2106   \texttt{\string\makeglossaries} or you have used
2107   \texttt{\string\nofiles}. If this is just a draft version of the
2108   document, you can suppress this message using the
2109   \texttt{nomissinggls} package option.%
2110 }
```

glossarywarning

```
2111 \newcommand*{@glsxtr@defaultnoglossarywarning}[1]{%
2112   \glossarysection[\glossarytoctitle]{\glossarytitle}
2113   \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glo@type @in\endcsname}
2114   \par
2115   \glsxtrifemptyglossary{#1}%
2116   {%
2117     \GlsXtrNoGlsWarningEmptyStart\space
2118     \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
2119     \medskip
2120     \noindent\texttt{\string\usepackage[nomain\ifglsacronym ,acronym\fi]%
2121       \glsopenbrace glossaries-extra\glsclosebrace}
2122     \medskip
2123     }%
2124     {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
2125   }%
2126   {%
2127     \IfFileExists{\jobname.\csname @glo@type @out\endcsname}
2128     {%
2129       \GlsXtrNoGlsWarningCheckFile
2130       {\jobname.\csname @glo@type @out\endcsname}
2131
2132       \ifglsautomake
2133
2134         \GlsXtrNoGlsWarningAutoMake{#1}
2135
2136       \else
2137
2138         \ifthenelse{\equal{#1}{main}}{%
2139           {%
2140             \GlsXtrNoGlsWarningEmptyMain\par
2141             \medskip
2142             \noindent\texttt{\string\usepackage[nomain]%
2143               \glsopenbrace glossaries-extra\glsclosebrace}
2144             \medskip
```

```

2145     }%
2146     {}%
2147
2148     \ifdefequal\makeglossaries\@no@makeglossaries
2149     {%
2150         \GlsXtrNoGlsWarningMisMatch
2151     }%
2152     {%
2153         \GlsXtrNoGlsWarningBuildInfo
2154     }%
2155     \fi
2156 }%
2157 {%
2158     \GlsXtrNoGlsWarningNoOut
2159     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
2160 }%
2161 }%
2162 \par
2163 \GlsXtrNoGlsWarningTail
2164 }

```

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.

```

2165 \ifpackageloaded{glossaries-accsupp}
2166 {

```

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

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

```

2167 \newcommand*\glsaccessname[1]{%
2168     \glsnameaccessdisplay
2169     {%
2170         \glsentryname{#1}%
2171     }%
2172     {#1}%
2173 }

```

\@glsname@ Redefine to use accessibility support.

```

2174 \def\@glsname#1#2[#3]{%
2175     \@gls@field@link{#1}{#2}{\glsaccessname{#2}#3}%
2176 }

```

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


```

2177 \newcommand*{\Glsaccessname}[1]{%
2178   \glsnameaccessdisplay
2179   {%
2180     \Glsentryname{#1}%
2181   }%
2182   {#1}%
2183 }

```

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

```

2184 \def\@Glsname@#1#2[#3]{%
2185   \@gls@field@link{#1}{#2}{\Glsaccessname{#2}#3}%
2186 }

```

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

```

2187 \newcommand*{\GLSaccessname}[1]{%
2188   \glsnameaccessdisplay
2189   {%
2190     \mfirstucMakeUppercase{\Glsentryname{#1}}%
2191   }%
2192   {#1}%
2193 }

```

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

```

2194 \def\@GLSname@#1#2[#3]{%
2195   \@gls@field@link{#1}{#2}{\GLSaccessname{#2}#3}%
2196 }

```

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

```

2197 \newcommand*{\glsaccesstext}[1]{%
2198   \glstextaccessdisplay
2199   {%
2200     \Glsentrytext{#1}%
2201   }%
2202   {#1}%
2203 }

```

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

```

2204 \def\@glstext@#1#2[#3]{%
2205   \@gls@field@link{#1}{#2}{\glsaccesstext{#2}#3}%
2206 }

```

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

```

2207 \newcommand*{\Glsaccesstext}[1]{%
2208   \glstextaccessdisplay
2209   {%
2210     \Glsentrytext{#1}%
2211   }%

```

```

2212     {#1}%
2213 }

```

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

```

2214 \def\@Glstext@#1#2[#3]{%
2215   \@gls@field@link{#1}{#2}{\Glsaccesstext{#2}#3}%
2216 }

```

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

```

2217   \newcommand*\Glsaccesstext}[1]{%
2218     \glstextaccessdisplay
2219     {%
2220       \mfirstucMakeUppercase{\glstrytext{#1}}%
2221     }%
2222     {#1}%
2223 }

```

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

```

2224 \def\@GLStext@#1#2[#3]{%
2225   \@gls@field@link{#1}{#2}{\GLSaccesstext{#2}#3}%
2226 }

```

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

```

2227   \newcommand*\glsaccessplural}[1]{%
2228     \glspluralaccessdisplay
2229     {%
2230       \glstryplural{#1}%
2231     }%
2232     {#1}%
2233 }

```

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

```

2234 \def\@glsplural@#1#2[#3]{%
2235   \@gls@field@link{#1}{#2}{\glsaccessplural{#2}#3}%
2236 }

```

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

```

2237   \newcommand*\GLsaccessplural}[1]{%
2238     \glspluralaccessdisplay
2239     {%
2240       \GLstryplural{#1}%
2241     }%
2242     {#1}%
2243 }

```

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

```

2244 \def\@GLsplural@#1#2[#3]{%

```

```

2245 \@gls@field@link{#1}{#2}{\Glsaccessplural{#2}#3}%
2246 }

```

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

```

2247 \newcommand*{\GLSaccessplural}[1]{%
2248 \glspluralaccessdisplay
2249 {%
2250 \mfirstucMakeUppercase{\glsentryplural{#1}}%
2251 }%
2252 {#1}%
2253 }

```

\@GLSplural@ Redefine to use accessibility support.

```

2254 \def\@GLSplural@#1#2[#3]{%
2255 \@gls@field@link{#1}{#2}{\GLSaccessplural{#2}#3}%
2256 }

```

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

```

2257 \newcommand*{\glsaccessfirst}[1]{%
2258 \glsfirstaccessdisplay
2259 {%
2260 \glsentryfirst{#1}%
2261 }%
2262 {#1}%
2263 }

```

\@glsfirst@ Redefine to use accessibility support.

```

2264 \def\@glsfirst@#1#2[#3]{%
2265 \@gls@field@link{#1}{#2}{\glsaccessfirst{#2}#3}%
2266 }

```

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

```

2267 \newcommand*{\Glsaccessfirst}[1]{%
2268 \glsfirstaccessdisplay
2269 {%
2270 \Glsentryfirst{#1}%
2271 }%
2272 {#1}%
2273 }

```

\@Glsfirst@ Redefine to use accessibility support.

```

2274 \def\@Glsfirst@#1#2[#3]{%
2275 \@gls@field@link{#1}{#2}{\Glsaccessfirst{#2}#3}%
2276 }

```

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

```

2277 \newcommand*{\GLSaccessfirst}[1]{%

```

```

2278 \glsfirstaccessdisplay
2279 {%
2280 \mfirstucMakeUppercase{\glsentryfirst{#1}}%
2281 }%
2282 {#1}%
2283 }

```

\@GLSfirst@ Redefine to use accessibility support.

```

2284 \def\@GLSfirst@#1#2[#3]{%
2285 \@gls@field@link{#1}{#2}{\GLSaccessfirst{#2}#3}%
2286 }

```

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

```

2287 \newcommand*{\glsaccessfirstplural}[1]{%
2288 \glsfirstpluralaccessdisplay
2289 {%
2290 \glsentryfirstplural{#1}%
2291 }%
2292 {#1}%
2293 }

```

glsfirstplural@ Redefine to use accessibility support.

```

2294 \def\@glsfirstplural@#1#2[#3]{%
2295 \@gls@field@link{#1}{#2}{\glsaccessfirstplural{#2}#3}%
2296 }

```

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

```

2297 \newcommand*{\Glsaccessfirstplural}[1]{%
2298 \glsfirstpluralaccessdisplay
2299 {%
2300 \Glsentryfirstplural{#1}%
2301 }%
2302 {#1}%
2303 }

```

Glsfirstplural@ Redefine to use accessibility support.

```

2304 \def\@Glsfirstplural@#1#2[#3]{%
2305 \@gls@field@link{#1}{#2}{\Glsaccessfirstplural{#2}#3}%
2306 }

```

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

```

2307 \newcommand*{\GLSaccessfirstplural}[1]{%
2308 \glsfirstpluralaccessdisplay
2309 {%
2310 \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
2311 }%
2312 {#1}%
2313 }

```

GLSfirstplural@ Redefine to use accessibility support.

```
2314 \def\@GLSfirstplural@#1#2[#3]{%
2315   \@gls@field@link{#1}{#2}{\GLSaccessfirstplural{#2}#3}%
2316 }
```

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

```
2317 \newcommand*\glsaccesssymbol[1]{%
2318   \glssymbolaccessdisplay
2319   {%
2320     \glsentrysymbol{#1}%
2321   }%
2322   {#1}%
2323 }
```

\@glssymbol@ Redefine to use accessibility support.

```
2324 \def\@glssymbol@#1#2[#3]{%
2325   \@gls@field@link{#1}{#2}{\glsaccesssymbol{#2}#3}%
2326 }
```

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

```
2327 \newcommand*\GLsaccesssymbol[1]{%
2328   \glssymbolaccessdisplay
2329   {%
2330     \GLsentrysymbol{#1}%
2331   }%
2332   {#1}%
2333 }
```

\@GLssymbol@ Redefine to use accessibility support.

```
2334 \def\@GLssymbol@#1#2[#3]{%
2335   \@gls@field@link{#1}{#2}{\GLsaccesssymbol{#2}#3}%
2336 }
```

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

```
2337 \newcommand*\GLSaccesssymbol[1]{%
2338   \glssymbolaccessdisplay
2339   {%
2340     \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
2341   }%
2342   {#1}%
2343 }
```

\@GLSsymbol@ Redefine to use accessibility support.

```
2344 \def\@GLSsymbol@#1#2[#3]{%
2345   \@gls@field@link{#1}{#2}{\GLSaccesssymbol{#2}#3}%
2346 }
```

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

```

2347 \newcommand*{\glsaccesssymbolplural}[1]{%
2348   \glssymbolpluralaccessdisplay
2349   {%
2350     \glsentrysymbolplural{#1}%
2351   }%
2352   {#1}%
2353 }
```

`\lssymbolplural@` Redefine to use accessibility support.

```

2354 \def\@glssymbolplural@#1#2[#3]{%
2355   \@gls@field@link{#1}{#2}{\glsaccesssymbolplural{#2}#3}%
2356 }
```

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

```

2357 \newcommand*{\Glsaccesssymbolplural}[1]{%
2358   \glssymbolpluralaccessdisplay
2359   {%
2360     \Glsentrysymbolplural{#1}%
2361   }%
2362   {#1}%
2363 }
```

`\GLsymbolplural@` Redefine to use accessibility support.

```

2364 \def\@GLsymbolplural@#1#2[#3]{%
2365   \@gls@field@link{#1}{#2}{\Glsaccesssymbolplural{#2}#3}%
2366 }
```

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

```

2367 \newcommand*{\GLSaccesssymbolplural}[1]{%
2368   \glssymbolpluralaccessdisplay
2369   {%
2370     \mfirstucMakeUppercase{\glsentrysymbolplural{#1}}%
2371   }%
2372   {#1}%
2373 }
```

`\GLSsymbolplural@` Redefine to use accessibility support.

```

2374 \def\@GLSsymbolplural@#1#2[#3]{%
2375   \@gls@field@link{#1}{#2}{\GLSaccesssymbolplural{#2}#3}%
2376 }
```

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

```

2377 \newcommand*{\glsaccessdesc}[1]{%
2378   \glsdescriptionaccessdisplay
2379   {%
2380     \glsentrydesc{#1}%

```

```

2381     }%
2382     {#1}%
2383 }

```

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

```

2384 \def\@glsdesc@#1#2[#3]{%
2385   \@gls@field@link{#1}{#2}{\glsaccessdesc{#2}#3}%
2386 }

```

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

```

2387 \newcommand*\Glsaccessdesc}[1]{%
2388   \glsdescriptionaccessdisplay
2389   {%
2390     \Glsentrydesc{#1}%
2391   }%
2392   {#1}%
2393 }

```

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

```

2394 \def\@Glsdesc@#1#2[#3]{%
2395   \@gls@field@link{#1}{#2}{\Glsaccessdesc{#2}#3}%
2396 }

```

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

```

2397 \newcommand*\GLSaccessdesc}[1]{%
2398   \glsdescriptionaccessdisplay
2399   {%
2400     \mfirstucMakeUppercase{\glsentrydesc{#1}}%
2401   }%
2402   {#1}%
2403 }

```

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

```

2404 \def\@GLSdesc@#1#2[#3]{%
2405   \@gls@field@link{#1}{#2}{\GLSaccessdesc{#2}#3}%
2406 }

```

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

```

2407 \newcommand*\glsaccessdescplural}[1]{%
2408   \glsdescriptionpluralaccessdisplay
2409   {%
2410     \glsentrydescplural{#1}%
2411   }%
2412   {#1}%
2413 }

```

```

@glsgdescplural@  Redefine to use accessibility support.
2414 \def\@glsgdescplural@#1#2[#3]{%
2415   \@gls@field@link{#1}{#2}{\glsaccessdescplural{#2}#3}%
2416 }

accessdescplural  Display the descplural value (no link and no check for existence) with the first letter converted
to upper case.
2417   \newcommand*{\Glsaccessdescplural}[1]{%
2418     \glsdescriptionpluralaccessdisplay
2419     {%
2420       \glentrydescplural{#1}%
2421     }%
2422     {#1}%
2423   }

@Glsdescplural@  Redefine to use accessibility support.
2424 \def\@Glsdescplural@#1#2[#3]{%
2425   \@gls@field@link{#1}{#2}{\Glsaccessdescplural{#2}#3}%
2426 }

Glsdescplural  Display the descplural value (no link and no check for existence) converted to upper case.
2427   \newcommand*{\GLSaccessdescplural}[1]{%
2428     \glsdescriptionpluralaccessdisplay
2429     {%
2430       \mfirstucMakeUppercase{\glentrydescplural{#1}}%
2431     }%
2432     {#1}%
2433   }

@GLSdescplural@  Redefine to use accessibility support.
2434 \def\@GLSdescplural@#1#2[#3]{%
2435   \@gls@field@link{#1}{#2}{\GLSaccessdescplural{#2}#3}%
2436 }

\glsaccessshort  Display the short form (no link and no check for existence).
2437   \newcommand*{\glsaccessshort}[1]{%
2438     \glsshortaccessdisplay
2439     {%
2440       \glentryshort{#1}%
2441     }%
2442     {#1}%
2443   }

\Glsaccessshort  Display the short form with first letter converted to uppercase (no link and no check for existence).
2444   \newcommand*{\Glsaccessshort}[1]{%
2445     \glsshortaccessdisplay
2446     {%

```



```

2447     \Glsentryshort{#1}%
2448   }%
2449   {#1}%
2450 }

```

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

```

2451 \newcommand*{\GLSaccessshort}[1]{%
2452   \glsshortaccessdisplay
2453   {%
2454     \mfirstucMakeUppercase{\glsentryshort{#1}}%
2455   }%
2456   {#1}%
2457 }

```

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

```

2458 \newcommand*{\lsaccessshortpl}[1]{%
2459   \glsshortpluralaccessdisplay
2460   {%
2461     \glsentryshortpl{#1}%
2462   }%
2463   {#1}%
2464 }

```

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

```

2465 \newcommand*{\lSaccessshortpl}[1]{%
2466   \glsshortpluralaccessdisplay
2467   {%
2468     \Glsentryshortpl{#1}%
2469   }%
2470   {#1}%
2471 }

```

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

```

2472 \newcommand*{\LSaccessshortpl}[1]{%
2473   \glsshortpluralaccessdisplay
2474   {%
2475     \mfirstucMakeUppercase{\glsentryshortpl{#1}}%
2476   }%
2477   {#1}%
2478 }

```

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

```

2479 \newcommand*{\glsaccesslong}[1]{%
2480   \glslongaccessdisplay{\glsentrylong{#1}}{#1}%
2481 }

```

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

```

2482
2483 \newcommand*{\Glsaccesslong}[1]{%
2484   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
2485 }

```

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

```

2486 \newcommand*{\Glsaccesslong}[1]{%
2487   \glslongaccessdisplay
2488   {%
2489     \mfirstucMakeUppercase{\Glsentrylong{#1}}%
2490   }%
2491   {#1}%
2492 }

```

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

```

2493 \newcommand*{\glsaccesslongpl}[1]{%
2494   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
2495 }

```

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

```

2496
2497 \newcommand*{\Glsaccesslongpl}[1]{%
2498   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
2499 }

```

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

```

2500 \newcommand*{\GLSaccesslongpl}[1]{%
2501   \glslongpluralaccessdisplay
2502   {%
2503     \mfirstucMakeUppercase{\Glsentrylongpl{#1}}%
2504   }%
2505   {#1}%
2506 }

```

End of if part

```

2507 }
2508 {

```

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

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

```

2509 \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.

```

2510 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

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

```

2511 \newcommand*{\GLSaccessname}[1]{%
2512   \protect\mfirstucMakeUppercase{\glsentryname{#1}}}

```

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

2513 `\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.

2514 `\newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}`

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

2515 `\newcommand*{\GLSaccesstext}[1]{%`

2516 `\protect\mfirstucMakeUppercase{\glsentrytext{#1}}}`

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

2517 `\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.

2518 `\newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}`

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

2519 `\newcommand*{\GLSaccessplural}[1]{%`

2520 `\protect\mfirstucMakeUppercase{\glsentryplural{#1}}}`

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

2521 `\newcommand*{\glsaccessfirst}[1]{\glsentryfirst{#1}}`

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

2522 `\newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}`

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

2523 `\newcommand*{\GLSaccessfirst}[1]{%`

2524 `\protect\mfirstucMakeUppercase{\glsentryfirst{#1}}}`

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

2525 `\newcommand*{\glsaccessfirstplural}[1]{\glsentryfirstplural{#1}}`

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

2526 `\newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}`

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

2527 `\newcommand*{\GLSaccessfirstplural}[1]{%`

2528 `\protect\mfirstucMakeUppercase{\glsentryfirstplural{#1}}}`

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

2529 `\newcommand*{\glsaccesssymbol}[1]{\glsentrysymbol{#1}}`

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

2530 `\newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}`

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

2531 `\newcommand*{\GLSaccesssymbol}[1]{%`
2532 `\protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}`

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

2533 `\newcommand*{\glaccesssymbolplural}[1]{\glentrysymbolplural{#1}}`

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

2534 `\newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}`

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

2535 `\newcommand*{\GLSaccesssymbolplural}[1]{%`
2536 `\protect\mfirstucMakeUppercase{\glentrysymbolplural{#1}}}`

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

2537 `\newcommand*{\glsaccessdesc}[1]{\glentrydesc{#1}}`

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

2538 `\newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}`

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

2539 `\newcommand*{\GLSaccessdesc}[1]{%`
2540 `\protect\mfirstucMakeUppercase{\glentrydesc{#1}}}`

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

2541 `\newcommand*{\glaccessdescplural}[1]{\glentrydescplural{#1}}`

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

2542 `\newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}`

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

2543 `\newcommand*{\GLSaccessdescplural}[1]{%`
2544 `\protect\mfirstucMakeUppercase{\glentrydescplural{#1}}}`

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

2545 `\newcommand*{\glsaccessshort}[1]{\glentryshort{#1}}`

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

2546 \newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}

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

2547 \newcommand*{\GLSaccessshort}[1]{%

2548 \protect\mfirstucMakeUppercase{\glsentryshort{#1}}}

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

2549 \newcommand*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}

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

2550 \newcommand*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}

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

2551 \newcommand*{\GLSaccessshortpl}[1]{%

2552 \protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}

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

2553 \newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}

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

2554 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}

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

2555 \newcommand*{\GLSaccesslong}[1]{%

2556 \protect\mfirstucMakeUppercase{\glsentrylong{#1}}}

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

2557 \newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}

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

2558 \newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}

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

2559 \newcommand*{\GLSaccesslongpl}[1]{%

2560 \protect\mfirstucMakeUppercase{\glsentrylongpl{#1}}}

End of else part

2561 }

1.5 Categories

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

```
2562 \glsaddstoragekey{category}{general}{\glscategory}
```

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

```
2563 \newcommand{\glsifcategory}[4]{%
2564   \ifglstfield{#1}{category}{#2}{#3}{#4}%
2565 }
```

Categories can have attributes.

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

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

```
2566 \newcommand*\glssetcategoryattribute[3]{%
2567   \csdef{@glstr@categoryattr@@#1@#2}{#3}%
2568 }
```

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

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

```
2569 \newcommand*\glsgetcategoryattribute[2]{%
2570   \csuse{@glstr@categoryattr@@#1@#2}%
2571 }
```

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

Tests if the category has the given attribute set.

```
2572 \newcommand*\glshascategoryattribute[4]{%
2573   \ifcvoid{@glstr@categoryattr@@#1@#2}{#4}{#3}%
2574 }
```

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

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

```
2575 \newcommand*\glssetattribute[3]{%
```

```

2576 \glsssetcategoryattribute{\glscategory{#1}}{#2}{#3}%
2577 }

```

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

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

```

2578 \newcommand*{\glsggetattribute}[2]{%
2579 \glsssetcategoryattribute{\glscategory{#1}}{#2}%
2580 }

```

`\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.

```

2581 \newcommand*{\glshasattribute}[4]{%
2582 \ifglsentryexists{#1}%
2583 {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
2584 {#4}%
2585 }

```

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

True if category has the attribute with the given value.

```

2586 \newcommand{\glssifcategoryattribute}[5]{%
2587 \ifcsundef{@glsxtr@categoryattr@#1@#2}%
2588 {#5}%
2589 {\ifcsstring{@glsxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
2590 }

```

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

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

```

2591 \newcommand{\glssifattribute}[5]{%
2592 \ifglsentryexists{#1}%
2593 {\glssifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
2594 {#5}%
2595 }

```

Set attributes for the default general category:

```
2596 \glsssetcategoryattribute{general}{regular}{true}
```

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

```
2597 \glsssetcategoryattribute{acronym}{regular}{true}
```

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

```
2598 \newcommand*{\glsssetregularcategory}[1]{%
2599   \glsssetcategoryattribute{#1}{regular}{true}%
2600 }
```

`\ifregularcategory` `\glssifregularcategory{<category>}{<true part>}{<false part>}`

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

```
2601 \newcommand{\glssifregularcategory}[3]{%
2602   \glssifcategoryattribute{#1}{regular}{true}{#2}{#3}%
2603 }
```

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

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

```
2604 \newcommand{\glssifregular}[3]{%
2605   \glssifregularcategory{\glscategory{#1}}{#2}{#3}%
2606 }
```

`\foreachincategory` `\glssforeachincategory[<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.

```
2607 \newcommand{\glssforeachincategory}[5][\@glo@types]{%
2608   \foralllglossaries[#1]{#3}%
2609   {%
2610     \forlgsentries[#3]{#4}%
2611     {%
2612       \glssifcategory{#4}{#2}{#5}{}%
2613     }%
2614   }%
2615 }
```


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.

```
2616 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%  
2617   \forallglossaries[#1]{#4}%  
2618   {%  
2619     \forallsentries[#4]{#5}%  
2620     {%  
2621       \glsifattribute{#5}{#2}{#3}{#6}{}%  
2622     }%  
2623   }%  
2624 }
```

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

```
2625 \ifdef\newterm  
2626 {%
```

`\newterm`

```
2627   \renewcommand*{\newterm}[2][ ]{%  
2628     \newglossaryentry{#2}%  
2629     {type={index},category=index,name={#2},%  
2630     description={\glstrpostdescription\nopostdesc},#1}%  
2631   }
```

Indexed terms are regular by default.

```
2632   \glsssetcategoryattribute{index}{regular}{true}
```

trpostdescindex

```
2633   \newcommand*{\glstrpostdescindex}{}  
2634 }  
2635 {}
```

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`.

```
2636 \ifdef\printsymbols  
2637 {%
```

glstrnewsymbol

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

```

2638 \newcommand*{\glxtrnewsymbol}[3][]{\%
2639   \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
2640 }

```

Symbols are regular by default.

```

2641 \glsssetcategoryattribute{symbol}{regular}{true}

```

rpostdescsymbol

```

2642 \newcommand*{\glxtrpostdescsymbol}{}

2643 }
2644 {}

```

Similar for the numbers option.

```

2645 \ifdef\printnumbers
2646 {%

```

glxtrnewnumber

```

2647 \ifdef\printnumbers
2648   \newcommand*{\glxtrnewnumber}[3][]{\%
2649     \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
2650   }

```

Numbers are regular by default.

```

2651 \glsssetcategoryattribute{number}{regular}{true}

```

rpostdescnumber

```

2652 \newcommand*{\glxtrpostdescnumber}{}

2653 }
2654 {}

```

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

```

2655 \newcommand*{\glxtrsetcategory}[2]{\%
2656   \@for\@glxtr@label:=#1\do
2657   {%
2658     \glsfieldxddef{\@glxtr@label}{category}{#2}%
2659   }%
2660 }

```

glxtrsetcategoryforall 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.

```

2661 \newcommand*{\glxtrsetcategoryforall}[2]{\%
2662   \forallglossaries[#1]{\@glxtr@type}{\%
2663     \forallsentries[\@glxtr@type]{\@glxtr@label}%
2664     {%
2665       \glsfieldxddef{\@glxtr@label}{category}{#2}%

```

```

2666     }%
2667 }%
2668 }

```

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

Apply title casing to the contents of the given field.

```

2669 \newcommand*{\glstrfieldtitlecase}[2]{%
2670   \expandafter\xcapitalisewords\expandafter
2671   {\csname glo@\glstetoklabel{#1}@#2\endcsname}%
2672 }

```

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 “firsttuc” convert first letter to upper case. If the attribute is “title” use title case.

```

2673 \@ifpackageloaded{glossaries-accsupp}
2674 {
2675   \renewcommand*{\glossentrydesc}[1]{%
2676     \glstoifexistsorwarn{#1}%
2677     {%
2678       \glsetabbrvfmt{\glscategory{#1}}%
2679       \glsoifattribute{#1}{glossdesc}{firsttuc}%
2680       {%
2681         \Glsaccessdesc{#1}%
2682       }%
2683     }%
2684     \glsoifattribute{#1}{glossdesc}{title}%
2685     {%
2686       \@glstr@do@titlecaps@warn
2687       \glsdescriptionaccessdisplay
2688       {%
2689         \glstrfieldtitlecase{#1}{desc}%
2690       }%
2691       {#1}%
2692     }%
2693     {%
2694       \glsoifattribute{#1}{glossdesc}{title}%
2695       {%
2696         \Glsaccessdesc{#1}%
2697       }%
2698     }%
2699 }
2700 {
2701   \renewcommand*{\glossentrydesc}[1]{%
2702     \glstoifexistsorwarn{#1}%

```

```

2703   {%
2704     \glsetabbrvfmt{\glscategory{#1}}%
2705     \gl@ifattribute{#1}{glossdesc}{firstuc}%
2706     {%
2707       \Glsentrydesc{#1}%
2708     }%
2709     {%
2710       \gl@ifattribute{#1}{glossdesc}{title}%
2711       {%
2712         \@glstr@do@titlecaps@warn
2713         \glstrfieldtitlecase{#1}{desc}%
2714       }%
2715       {%
2716         \Glsentrydesc{#1}%
2717       }%
2718     }%
2719   }%
2720 }
2721 }

```

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

```

2722 \@ifpackageloaded{glossaries-accsupp}
2723 {
2724   \renewcommand*{\glossentryname}[1]{%
2725     \gl@ifexistsorwarn{#1}%
2726     {%
2727       \glsetabbrvfmt{\glscategory{#1}}%
2728       \gl@ifattribute{#1}{glossname}{firstuc}%
2729       {%
2730         \glsnameaccessdisplay
2731         {%
2732           \glsnamefont{\Glsentryname{#1}}%
2733         }%
2734         {#1}%
2735       }%
2736       {%
2737         \gl@ifattribute{#1}{glossname}{title}%
2738         {%
2739           \@glstr@do@titlecaps@warn
2740           \glsnameaccessdisplay
2741           {%
2742             \glsnamefont{\glstrfieldtitlecase{#1}{name}}%
2743           }%
2744           {#1}%
2745         }%
2746         {%
2747           \gl@ifattribute{#1}{glossname}{uc}%
2748           {%

```

```

2749         \glsnameaccessdisplay
2750     {%
    Hide the label from the upper-casing command.
2751         \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
2752         \glsnamefont{\mfirstucMakeUppercase{\glo@name}}%
2753     }%
2754     {#1}%
2755 }%
2756 {%
2757     \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
2758     \glsnameaccessdisplay
2759     {%
2760         \expandafter\glsnamefont\expandafter{\glo@name}%
2761     }%
2762     {#1}%
2763 }%
2764 }%
2765 }%

```

Do post-name hook:

```

2766     \glstrpostnamehook{#1}%
2767 }%
2768 }
2769 }
2770 {
2771     \renewcommand*{\glossentryname}[1]{%
2772         \glsdoifexistsorwarn{#1}%
2773         {%
2774             \glsssetabbrvfmt{\glscategory{#1}}%
2775             \glsifattribute{#1}{glossname}{firstuc}%
2776             {%
2777                 \glsnamefont{\Glsentryname{#1}}%
2778             }%
2779             {%
2780                 \glsifattribute{#1}{glossname}{title}%
2781                 {%
2782                     \@glstr@do@titlecaps@warn
2783                     \glsnamefont{\glstrfieldtitlecase{#1}{name}}%
2784                 }%
2785                 {%
2786                     \glsifattribute{#1}{glossname}{uc}%
2787                     {%

```

Hide the label from the upper-casing command.

```

2788         \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
2789         \glsnamefont{\mfirstucMakeUppercase{\glo@name}}%
2790     }%
2791     {%

```

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.

```

2792      \letcs{\glo@name}{\glo@glstetoklabel{#1}@name}%
2793      \expandafter\glsnamefont\expandafter{\glo@name}%
2794      }%
2795      }%
2796      }%
2797      }%

```

Do post-name hook:

```

2798      \glstrpostnamehook{#1}%
2799  }
2800 }

```

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

```

2801 \ifpackageloaded{glossaries-accsupp}
2802 {
2803   \renewcommand*{\Glossentryname}[1]{%
2804     \glsdoifexistsorwarn{#1}%
2805     {%
2806       \glssetabbrvfmt{\glscategory{#1}}%
2807       \glsnameaccessdisplay
2808       {%
2809         \glsnamefont{\Glsentryname{#1}}%
2810       }%
2811       {#1}%

```

Do post-name hook:

```

2812       \glstrpostnamehook{#1}%
2813     }%
2814   }
2815 }
2816 {
2817   \renewcommand*{\Glossentryname}[1]{%
2818     \glsdoifexistsorwarn{#1}%
2819     {%
2820       \glssetabbrvfmt{\glscategory{#1}}%
2821       \glsnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

2822       \glstrpostnamehook{#1}%
2823     }%
2824   }
2825 }

```

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

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

```

2826 \newcommand*{\glstrpostnamehook}[1]{%

```

```

2827 \def\@glsnumberformat{glsnumberformat}%
2828 \glsxtrdoautoindexname{#1}{indexname}%
2829 }

```

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

```

2830 \newif\if@glsxtr@format@override
2831 \@glsxtr@format@overridefalse

```

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

`xFormatOverride`

```

2832 \@ifpackageloaded{hyperref}
2833 {

```

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

```

2834 \ifHy@hyperindex
2835 \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
2836 \@glsxtr@format@override true
2837 \appto\theindex{\let\glsnumber\@firstofone}%
2838 }
2839 \else
2840 \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
2841 \@glsxtr@format@override true
2842 \appto\theindex{\let\glsnumber\hyperpage}%
2843 }
2844 \fi
2845 }
2846 {
2847 \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
2848 \@glsxtr@format@override true
2849 }
2850 }
2851 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

`doautoindexname`

```

2852 \newcommand*{\glsxtrdoautoindexname}[2]{%
2853 \glsasattribute{#1}{#2}%
2854 {%

```

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.

```

2855 \@glsxtr@autoindex@setname{#1}%

```

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

```

2856 \protected@edef\@glsxtr@attrval{\glsgetattribute{#1}{#2}}%
2857 \if@glsxtr@format@override
2858 \ifdefstring{\@glsnumberformat}{glsnumberformat}{}%
2859 {\let\@glsxtr@attrval\@glsnumberformat}%

```

```

2860 \fi
2861 \ifdefstring{\@glsxtr@attrval}{true}%
2862 {}%
2863 {\eappto\@glo@name{\@glsxtr@autoindex@encap\@glsxtr@attrval}}%
2864 \expandafter\index\expandafter{\@glo@name}%
2865 }%
2866 {}%
2867 }

```

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

```

2868 \newcommand*{\@glsxtr@autoindex@setname}[1]{%
2869 \def\@glo@name{\string\glsentryname{#1}}%
2870 \glsletentryfield{\@glo@sort}{#1}{sort}%
2871 \@gls@checkmkidxchars\@glo@sort
2872 \@glsxtr@autoindex@doextra@esc\@glo@sort
2873 \epreto\@glo@name{\@glo@sort\@glsxtr@autoindex@at}%
2874 }

```

dex@doextra@esc

```

2875 \newcommand*{\@glsxtr@autoindex@doextra@esc}[1]{%
    Escape the escape character unless it has already been escaped.
2876 \ifx\@glsxtr@autoindex@esc\@gls@quotechar
2877 \else
2878 \def\@gls@checkedmkidx{}%
2879 \edef\@glsxtr@checkspch{%
2880 \noexpand\@glsxtr@autoindex@escquote\expandonce{#1}%
2881 \noexpand\@empty\@glsxtr@autoindex@esc\noexpand\@nnil
2882 \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%
2883 @@glsxtr@checkspch
2884 \let#1\@gls@checkedmkidx\relax
2885 \fi

```

Escape actual character unless it has already been escaped.

```

2886 \ifx\@glsxtr@autoindex@at\@gls@actualchar
2887 \else
2888 \def\@gls@checkedmkidx{}%
2889 \edef\@glsxtr@checkspch{%
2890 \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
2891 \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
2892 \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
2893 @@glsxtr@checkspch
2894 \let#1\@gls@checkedmkidx\relax
2895 \fi

```

Escape level character unless it has already been escaped.

```

2896 \ifx\@glsxtr@autoindex@level\@gls@levelchar
2897 \else
2898 \def\@gls@checkedmkidx{}%
2899 \edef\@glsxtr@checkspch{%

```



```

2900 \noexpand\@glxtr@autoindex@esclevel\expandonce{#1}%
2901 \noexpand\@empty\@glxtr@autoindex@level\noexpand\@nnil
2902 \@glxtr@autoindex@level\noexpand\@empty\noexpand\@glxtr@endescspch}%
2903 \@@glxtr@checkspch
2904 \let#1\@glxtr@checkedmkidx\relax
2905 \fi

```

Escape encap character unless it has already been escaped.

```

2906 \ifx\@glxtr@autoindex@encap\@glxtr@encapchar
2907 \else
2908 \def\@glxtr@checkedmkidx{}%
2909 \edef\@@glxtr@checkspch{%
2910 \noexpand\@glxtr@autoindex@escencap\expandonce{#1}%
2911 \noexpand\@empty\@glxtr@autoindex@encap\noexpand\@nnil
2912 \@glxtr@autoindex@encap\noexpand\@empty\noexpand\@glxtr@endescspch}%
2913 \@@glxtr@checkspch
2914 \let#1\@glxtr@checkedmkidx\relax
2915 \fi
2916 }

```

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.

```

2917 \newcommand*{\@glxtr@autoindex@at}{}

```

trSetActualChar Set the actual character.

```

2918 \newcommand*{\GlsXtrSetActualChar}[1]{%
2919 \gdef\@glxtr@autoindex@at{#1}%
2920 \def\@glxtr@autoindex@escat##1#1##2#1##3\@glxtr@endescspch{%
2921 \@@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escat}{##1}{##2}{##3}%
2922 }%
2923 }
2924 \@onlypreamble\GlsXtrSetActualChar
2925 \makeatother
2926 \GlsXtrSetActualChar{@}
2927 \makeatletter

```

autoindex@encap Encap character for use with \index.

```

2928 \newcommand*{\@glxtr@autoindex@encap}{}

```

XtrSetEncapChar Set the encap character.

```

2929 \newcommand*{\GlsXtrSetEncapChar}[1]{%
2930 \gdef\@glxtr@autoindex@encap{#1}%
2931 \def\@glxtr@autoindex@escencap##1#1##2#1##3\@glxtr@endescspch{%
2932 \@@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escencap}{##1}{##2}{##3}%
2933 }%
2934 }
2935 \GlsXtrSetEncapChar{||}
2936 \@onlypreamble\GlsXtrSetEncapChar

```

autoindex@level Level character for use with \index.

```
2937 \newcommand*{\@glxtr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
2938 \newcommand*{\GlsXtrSetLevelChar}[1]{%
2939   \gdef\@glxtr@autoindex@level{#1}%
2940   \def\@glxtr@autoindex@esclevel##1#1##2#1##3\@glxtr@endescspch{%
2941     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@esclevel}{##1}{##2}{##3}%
2942   }%
2943 }
2944 \GlsXtrSetLevelChar{!}
2945 \@onlypreamble\GlsXtrSetLevelChar
```

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

```
2946 \newcommand*{\@glxtr@autoindex@esc}{}
```

lsXtrSetEscChar Set the escape character.

```
2947 \newcommand*{\GlsXtrSetEscChar}[1]{%
2948   \gdef\@glxtr@autoindex@esc{#1}%
2949   \def\@glxtr@autoindex@escquote##1#1##2#1##3\@glxtr@endescspch{%
2950     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escquote}{##1}{##2}{##3}%
2951   }%
2952 }
2953 \GlsXtrSetEscChar{"}
2954 \@onlypreamble\GlsXtrSetEscChar
```

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

```
2955 \ifdef\actualchar
2956 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
2957 {}
```

Quote character \quotechar:

```
2958 \ifdef\quotechar
2959 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
2960 {}
```

Level character \levelchar:

```
2961 \ifdef\levelchar
2962 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
2963 {}
```

Encap character \encapchar:

```
2964 \ifdef\encapchar
2965 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
2966 {}
```

leto@endescspch

```
2967 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

toindex@esc@spch

$\@@\glstr@autoindex@escspch\{\langle char \rangle\}\{\langle cs \rangle\}\{\langle pre \rangle\}\{\langle mid \rangle\}\{\langle post \rangle\}$

```

2968 \newcommand*{\@@glstr@autoindex@escspch}[5]{%
2969   \@glstmpb=\expandafter{\@glsccheckedmkidx}%
2970   \toks@={#3}%
2971   \ifx\@nnil#3\relax
2972     \def\@@glstr@checkspch{\@glstr@gobbleto@endescspch#5\@glstr@endescspch}%
2973   \else
2974     \ifx\@nnil#4\relax
2975       \edef\@glsccheckedmkidx{\the\@glstmpb\the\toks@}%
2976       \def\@@glstr@checkspch{\@glstr@gobbleto@endescspch
2977         #4#5\@glstr@endescspch}%
2978     \else
2979       \edef\@glsccheckedmkidx{\the\@glstmpb\the\toks@
2980         \@glstr@autoindex@esc#1}%
2981       \def\@@glstr@checkspch{#2#5#1\@nnil#1\@glstr@endescspch}%
2982     \fi
2983   \fi
2984   \@@glstr@checkspch
2985 }

```

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

```

2986 \renewcommand*{\Glossentrydesc}[1]{%
2987   \glsdofexistsorwarn{#1}%
2988   {%
2989     \glsssetabbrvfmt{\glscategory{#1}}%
2990     \Glsaccessdesc{#1}%
2991   }%
2992 }

```

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

```

2993 \renewcommand*{\lossentrysymbol}[1]{%
2994   \glsdofexistsorwarn{#1}%
2995   {%
2996     \glsssetabbrvfmt{\glscategory{#1}}%
2997     \Glsaccesssymbol{#1}%
2998   }%
2999 }

```

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

```

3000 \renewcommand*{\Glossentrysymbol}[1]{%
3001   \glsdofexistsorwarn{#1}%
3002   {%
3003     \glsssetabbrvfmt{\glscategory{#1}}%
3004     \Glsaccesssymbol{#1}%
3005   }%
3006 }

```

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

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

```
3007 \newcommand*{\GlsXtrEnableInitialTagging}{%
3008   \ifstar\s@glsxtr@enabletagging\@glsxtr@enabletagging
3009 }
3010 \@onlypreamble\GlsXtrEnableInitialTagging
```

`r@enabletagging` Starred version undefines command.

```
3011 \newcommand*{\s@glsxtr@enabletagging}[2]{%
3012   \undef#2%
3013   \@glsxtr@enabletagging{#1}{#2}%
3014 }
```

`r@enabletagging` Internal command.

```
3015 \newcommand*{\@glsxtr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
3016   \@for\@glsxtr@cat:=#1\do
3017   {%
3018     \ifdefempty\@glsxtr@cat
3019     {}%
3020     {\glsssetcategoryattribute{\@glsxtr@cat}{tagging}{true}}%
3021   }%
3022   \newrobustcmd*#2[1]{##1}%
3023   \def\@glsxtr@taggingcs{#2}%
3024   \renewcommand*\@glsxtr@activate@initialtagging{%
3025     \let#2\@glsxtr@tag
3026   }%
3027   \ifundef\@gls@preglossaryhook
3028   {\GlossariesExtraWarning{Initial tagging requires at least
3029     glossaries.sty v4.19 to work correctly}}%
3030   {}%
3031 }
```

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.

`fu@checkword@do` If this command hasn't been defined, then we have pre v2.02 of `mfirstuc`

```
3032 \ifundef\mfu@checkword@do
3033 {
3034   \newcommand*{\mfu@checkword@do}[1]{%
3035     \ifdefstring{\mfu@checkword@arg}{#1}%
3036     {%
3037       \let\@mfu@domakefirstuc\@firstofone
3038       \listbreak
3039     }%
3040   }
```

```

3040 {}%
3041 }

```

`\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.

```

3042 \ifundef\mfu@checkword
3043 {
3044   \newcommand{\@glstr@do@titlecaps@warn}{%
3045     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
3046       support not available}%
3047     \let\@glstr@do@titlecaps@warn\relax
3048   }
3049 }
3050 {
3051   \renewcommand*\mfu@checkword[1]{%
3052     \def\mfu@checkword@arg{#1}%
3053     \let\@mfu@domakefirstuc\makefirstuc
3054     \forlistloop\mfu@checkword@do\@mfu@nocaplist
3055   }
3056 }
3057 }
3058 {}% no patch required

```

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

```

3059 \newcommand*\@glstr@do@titlecaps@warn{}

```

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

```

3060 \newcommand*\@glstr@activate@initialtagging{}

```

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

```

3061 \newrobustcmd*\@glstr@tag[1]{%
3062   \glstr@attribute{\glstr@currententrylabel}{tagging}{true}%
3063   {\glstr@tagfont{#1}}{#1}%
3064 }

```

`\glstr@tagfont` Used in the glossary.

```

3065 \newcommand*\glstr@tagfont[1]{\underline{#1}}

```

`preglossaryhook` This macro was introduced in `glossaries` version 4.19, so it may not be defined. If it hasn't been defined this feature is unavailable.

```

3066 \ifdef\@glstr@preglossaryhook
3067 {
3068   \renewcommand*\@glstr@preglossaryhook{%
3069     \@glstr@activate@initialtagging
3070     \let\@glstr@org@postdescription\glstr@postdescription
3071     \renewcommand*\glstr@postdescription{%

```

```

3072     \glsxtrpostdescription
3073     \@glsxtr@org@postdescription
3074 }%
3075 }%
3076 }
3077 {}

```

`postdescription` This command will only be used if `\@gls@preglossaryhook` is available *and* the glossary style uses `\glspostdescription` without modifying it. (`\nopostdesc` will suppress this.) The glossaries-extra-stylemods package will add the post description hook to all the predefined styles that don't include it.

```

3078 \newcommand*{\glsxtrpostdescription}{%
3079   \csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%
3080 }

```

`postdescgeneral`

```

3081 \newcommand*{\glsxtrpostdescgeneral}{}

```

`xtrpostdescterm`

```

3082 \newcommand*{\glsxtrpostdescterm}{}

```

`postdescacronym`

```

3083 \newcommand*{\glsxtrpostdescacronym}{}

```

`descabbreviation`

```

3084 \newcommand*{\glsxtrpostdescabbreviation}{}

```

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

```

3085 \renewcommand*{\glspostlinkhook}{%
3086   \ifglentryexists{\glslabel}{\glsxtrpostlinkhook}{}%
3087 }

```

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

```

3088 \newcommand*{\glsxtrpostlinkhook}{%
3089   \glsxtrdiscardperiod{\glslabel}%
3090   {\glsxtrpostlinkendsentence}%
3091   {\glsxtrpostlink}%
3092 }

```

`\glsxtrpostlink`

```

3093 \newcommand*{\glsxtrpostlink}{%
3094   \csuse{glsxtrpostlink\glscategory{\glslabel}}%
3095 }

```

linkendsentence Done by \glxstrpostlinkhook if a full stop is discarded.

```
3096 \newcommand*{\glxstrpostlinkendsentence}{%
3097   \ifcsdef{glxstrpostlink\glscategory{\glslabel}}
3098   {%
3099     \csuse{glxstrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
3100   .\spacefactor\sfcode'\. \relax
3101 }%
3102 {%
```

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

```
3103   \spacefactor\sfcode'\. \relax
3104 }%
3105 }
```

DescOnFirstUse 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.

```
3106 \newcommand*{\glxstrpostlinkAddDescOnFirstUse}{%
3107   \glxstrifwasfirstuse{\space(\glssaccessdesc{\glslabel})}{}%
3108 }
```

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.

```
3109 \newcommand*{\glxstrpostlinkAddSymbolOnFirstUse}{%
3110   \glxstrifwasfirstuse
3111   {%
3112     \ifglshassymbol{\glslabel}{\space(\glssaccesssymbol{\glslabel})}{}%
3113   }%
3114   {}%
3115 }
```

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).

```
3116 \newcommand*{\glxstrdiscardperiod}[3]{%
3117   \glxstrifwasfirstuse
3118   {%
3119     \glusifattribute{#1}{retainfirstuseperiod}{true}%
3120     {#3}%
3121   }%
3122   \glusifattribute{#1}{discardperiod}{true}%
3123   {%
3124     \glusifplural
3125     {%
3126       \glusifattribute{#1}{pluraldiscardperiod}{true}%
3127       {\glxstrifperiod{#2}{#3}}%

```

```

3128         {#3}%
3129     }%
3130     {%
3131         \glxstrifperiod{#2}{#3}%
3132     }%
3133 }%
3134 {#3}%
3135 }%
3136 }%
3137 {%
3138     \glusifattribute{#1}{discardperiod}{true}%
3139     {%
3140         \glusifplural
3141         {%
3142             \glusifattribute{#1}{pluraldiscardperiod}{true}%
3143             {\glxstrifperiod{#2}{#3}}%
3144             {#3}%
3145         }%
3146         {%
3147             \glxstrifperiod{#2}{#3}%
3148         }%
3149     }%
3150     {#3}%
3151 }%
3152 }

```

`\glxstrifperiod` 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`

```

3153 \newcommand*{\glxstrifperiod}[1]{\new@ifnextchar.\{\@firstoftwo{#1}\}}

```

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

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

```

3154 \newcommand*{\glxtr@punclist}{.,:;?!}

```

`punctuationmark` Add character to punctuation list.

```

3155 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}

```

`punctuationmarks` Reset the punctuation list.

```

3156 \newcommand*{\glxtrsetpunctuationmarks}[1]{\def\glxtr@punclist{#1}}

```

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

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

```

3157 \newcommand*{\glxstrifnextpunc}[2]{%

```



```

3158 \def\reserved@a{#1}%
3159 \def\reserved@b{#2}%
3160 \futurelet\@glspunc@token\glxtr@ifnextpunc
3161 }

```

`glxtr@ifnextpunc`

```

3162 \newcommand*\glxtr@ifnextpunc{%
3163 \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
3164 \reserved@b
3165 }

```

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

```

3166 \newcommand*\glxtr@ifpunctoken[1]{%
3167 \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
3168 }

```

`glxtr@ifpunctoken`

```

3169 \def\@glxtr@ifpunctoken#1#2{%
3170 \let\reserved@d=#2%
3171 \ifx\reserved@d\@nnil
3172 \let\glxtr@next\@glxtr@notfoundinlist
3173 \else
3174 \ifx#1\reserved@d
3175 \let\glxtr@next\@glxtr@foundinlist
3176 \else
3177 \let\glxtr@next\@glxtr@ifpunctoken
3178 \fi
3179 \fi
3180 \glxtr@next#1%
3181 }

```

`glxtr@foundinlist`

```

3182 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}

```

`glxtr@notfoundinlist`

```

3183 \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.

```

3184 \newcommand*\glxtr@dopostpunc[1]{%
3185 \glxtr@ifnextpunc{\@glxtr@swaptwo{#1}}{#1}%
3186 }

```

`glxtr@swaptwo`

```

3187 \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.

```
3188 \define@key{glstrabbrv}{category}{%
3189   \edef\glscategorylabel{#1}%
3190   \ifcsdef{@glssabbrv@current@#1}%
3191     {%
3192       \glstr@applyabbrvstyle{\csname @glssabbrv@current@#1\endcsname}%
3193     }%
3194   {%}%
3195 }
```

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

```
3196 \define@key{glstrabbrv}{shortplural}{%
3197   \def\@glss@shortpl{#1}%
3198 }
```

Similarly for the long plural form.

```
3199 \define@key{glstrabbrv}{longplural}{%
3200   \def\@glss@longpl{#1}%
3201 }
```

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

\glsshortpltok

```
3202 \newtoks\glsshortpltok
```

\glslongpltok

```
3203 \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.

```
3204 \newcommand*{\@glstr@insertdots}[2]{%
3205   \def#1{%
3206     \@glstr@insert@dots#1#2\@nnil
3207   }
```

glstr@insert@dots

```
3208 \newcommand*{\@glstr@insert@dots}[2]{%
3209   \ifx\@nnil#2\relax
```

```

3210 \let\@glsxtr@insert@dots@next\@gobble
3211 \else
3212 \ifx\relax#2\relax
3213 \else
3214 \appto#1{#2.}%
3215 \fi
3216 \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
3217 \fi
3218 \@glsxtr@insert@dots@next#1%
3219 }

```

newabbreviation Define a new generic abbreviation.

```

3220 \newcommand*{\newabbreviation}[4][]{%
3221 \glskeylisttok{#1}%
3222 \glslabeltok{#2}%
3223 \glsshorttok{#3}%
3224 \glslongtok{#4}%

  Get the category.
3225 \def\glscategorylabel{abbreviation}%
3226 \glsxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
3227 \setkeys*{glsxtrabbrv}[shortplural,longplural]{#1}%

  Set the default long plural
3228 \def\@gls@longpl{#4\glspluralsuffix}%

  Has the insertdots attribute been set?
3229 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
3230 {%
3231 \@glsxtr@insertdots\@gls@short{#3}%
3232 \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
3233 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
3234 {%
3235 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
3236 '\abbrvpluralsuffix}%
3237 }%
3238 {%
3239 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
3240 {%
3241 \let\@gls@shortpl\@gls@short
3242 }%
3243 {%
3244 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
3245 \abbrvpluralsuffix}%
3246 }%
3247 }%
3248 }%
3249 {%

  insertdots not true.
3250 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%

```

```

3251   {%
3252     \def\@gls@shortpl{#3'\abbrvpluralsuffix}%
3253   }%
3254   {%
3255     \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
3256     {%
3257       \def\@gls@shortpl{#3}%
3258     }%
3259     {%
3260       \def\@gls@shortpl{#3\abbrvpluralsuffix}%
3261     }%
3262   }%
3263 }%

```

Hook for further customisation if required:

```

3264 \glsxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%

```

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

```

3265 \setkeys*{\glsxtrabbrv}[category]{#1}%

```

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

```

3266 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
3267 \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

3268 \newabbreviationhook

```

Define this entry:

```

3269 \protected@edef\@do@newglossaryentry{%
3270   \noexpand\newglossaryentry{\the\glslabeltok}%
3271   {%
3272     type=\glsxtrabbrvtype,%
3273     category=abbreviation,%
3274     short={\the\glsshorttok},%
3275     shortplural={\the\glsshortpltok},%
3276     long={\the\glslongtok},%
3277     longplural={\the\glslongpltok},%
3278     name={\the\glsshorttok},%
3279     \CustomAbbreviationFields,%
3280     \the\glskeylisttok
3281   }%
3282 }%
3283 \@do@newglossaryentry
3284 \GlsXtrPostNewAbbreviation
3285 }

```

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

```

3286 \newcommand*{\glsxtrnewabbrevpresetkeyhook}[3]{}

```

`\NewAbbreviation` Hook used by abbreviation styles.

```

3287 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

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

```
3288 \newcommand*{\newabbreviationhook}{}
```

`\CustomAbbreviationFields`

```
3289 \newcommand*{\CustomAbbreviationFields}{}
```

`\glstrfullformat` Full format without case change.

```
3290 \newcommand*{\glstrfullformat}[2]{%
3291   \glstrfirstlongfont{\glstraccesslong{#1}}#2\glstrfullsep{#1}%
3292   (\protect\glstrfirstabbrvfont{\glstraccessshort{#1}})%
3293 }
```

`\Glsstrfullformat` Full format with case change.

```
3294 \newcommand*{\Glsstrfullformat}[2]{%
3295   \glstrfirstlongfont{\Glsstraccesslong{#1}}#2\glstrfullsep{#1}%
3296   (\protect\glstrfirstabbrvfont{\Glsstraccessshort{#1}})%
3297 }
```

`\glstrfullplformat` Plural full format without case change.

```
3298 \newcommand*{\glstrfullplformat}[2]{%
3299   \glstrfirstlongfont{\glstraccesslongpl{#1}}#2\glstrfullsep{#1}%
3300   (\protect\glstrfirstabbrvfont{\glstraccessshortpl{#1}})%
3301 }
```

`\Glsstrfullplformat` Plural full format with case change.

```
3302 \newcommand*{\Glsstrfullplformat}[2]{%
3303   \glstrfirstlongfont{\Glsstraccesslongpl{#1}}#2\glstrfullsep{#1}%
3304   (\protect\glstrfirstabbrvfont{\Glsstraccessshortpl{#1}})%
3305 }
```

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

```
3306 \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.

```
3307 \newcommand*{\glstrinlinefullformat}{\glstrfullformat}
```

`\Glsstrinlinefullformat` Full format with case change.

```
3308 \newcommand*{\Glsstrinlinefullformat}{\Glsstrfullformat}
```

`\glstrfullplformat` Plural full format without case change.

```
3309 \newcommand*{\glstrinlinefullplformat}{\glstrfullplformat}
```

`\Glsstrinlinefullplformat` Plural full format with case change.

```
3310 \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
3311 \renewcommand*{\glsentryfull}[1]{\glsxtrinlinefullformat{#1}{}}

\Glsentryfull
3312 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}

\glsentryfullpl
3313 \renewcommand*{\glsentryfullpl}[1]{\glsxtrinlinefullplformat{#1}{}}

\Glsentryfullpl
3314 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}

\firstabbrvfont  Font changing command used for the abbreviation on first use or in the full format.
3315 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}

\abbrvdefaultfont  Font changing command used for the abbreviation on first use or in the full format.
3316 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}

\glsabbrvfont  Font changing command used for the abbreviation on subsequent use.
3317 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}

\abbrvdefaultfont
3318 \newcommand*{\glsabbrvdefaultfont}[1]{#1}

\firstlongfont  Font changing command used for the long form on first use or in the full format.
3319 \newcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{#1}}

\longdefaultfont
3320 \newcommand*{\glsfirstlongdefaultfont}[1]{#1}

\brvpluralsuffix  Default plural suffix.
3321 \newcommand*{\abbrvpluralsuffix}{\glspluralsuffix}

\glsxtrfull  Full form (no case-change).
3322 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\ns@glsxtrfull}
3323 \newcommand*{\ns@glsxtrfull}[2][{}]{%
3324   \new@ifnextchar[{\@glsxtr@full{#1}{#2}}]{%
3325     {\@glsxtr@full{#1}{#2}[]}%
3326 }

```

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

```

3327 \def\@glxtr@full#1#2[#3]{%
3328   \glsdoifexists{#2}%
3329   {%
3330     \glsetabbrvfmt{\glscategory{#2}}%
3331     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3332     \let\glsifplural\@secondoftwo
3333     \let\glscapscase\@firstofthree
3334     \let\glsinsert\@empty
3335     \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.

```

3336   \glxtrsetupfulldefs
3337   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3338   }%
3339   \glspostlinkhook
3340 }

```

`trsetupfulldefs`

```

3341 \newcommand*{\glxtrsetupfulldefs}{%
3342   \let\glxtrifwasfirstuse\@firstoftwo
3343 }

```

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

```

3344 \newrobustcmd*{\Glsxtrfull}{\@gl@hyp@opt\ns@Glsxtrfull}
3345 \newcommand*\ns@Glsxtrfull[2][ ]{%
3346   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}%
3347   {\@Glsxtr@full{#1}{#2} [ ]}%
3348 }

```

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

```

3349 \def\@Glsxtr@full#1#2[#3]{%
3350   \glsdoifexists{#2}%
3351   {%
3352     \glsetabbrvfmt{\glscategory{#2}}%
3353     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3354     \let\glsifplural\@secondoftwo
3355     \let\glscapscase\@secondofthree
3356     \let\glsinsert\@empty
3357     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
3358     \glxtrsetupfulldefs
3359     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3360   }%
3361   \glspostlinkhook
3362 }

```

`\GLSxtrfull` Full form (all uppercase).

```
3363 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\ns@GLSxtrfull}
3364 \newcommand*\ns@GLSxtrfull[2][\%
3365   \new@ifnextchar[\@GLSxtr@full{#1}{#2}]{%
3366     {\@GLSxtr@full{#1}{#2}[]}%
3367 }
```

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

```
3368 \def\@GLSxtr@full#1#2[#3]{%
3369   \glsdoifexists{#2}%
3370   {%
3371     \glssetabbrvfmt{\glscategory{#2}}%
3372     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3373     \let\glsifplural\@secondoftwo
3374     \let\glsapscase\@thirdofthree
3375     \let\glsinsert\@empty
3376     \def\glscustomtext{\mfirstucMakeUppercase{\glsxtrinlinefullformat{#2}{#3}}}%
3377     \glsxtrsetupfulldefs
3378     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3379   }%
3380   \glspostlinkhook
3381 }
```

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

```
3382 \newrobustcmd*{\glsxtrfullpl}{\@gls@hyp@opt\ns@glsxtrfullpl}
3383 \newcommand*\ns@glsxtrfullpl[2][\%
3384   \new@ifnextchar[\@glsxtr@fullpl{#1}{#2}]{%
3385     {\@glsxtr@fullpl{#1}{#2}[]}%
3386 }
```

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

```
3387 \def\@glsxtr@fullpl#1#2[#3]{%
3388   \glsdoifexists{#2}%
3389   {%
3390     \glssetabbrvfmt{\glscategory{#2}}%
3391     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3392     \let\glsifplural\@firstoftwo
3393     \let\glsapscase\@firstofthree
3394     \let\glsinsert\@empty
3395     \def\glscustomtext{\glsxtrinlinefullplformat{#2}{#3}}%
3396     \glsxtrsetupfulldefs
3397     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3398   }%
3399   \glspostlinkhook
3400 }
```

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

```
3401 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\ns@Glsxtrfullpl}
3402 \newcommand*\ns@Glsxtrfullpl[2][\%
```



```

3403 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
3404           {\@Glsxtr@fullpl{#1}{#2}[]}%
3405 }

```

\@Glsxtr@fullpl Low-level macro:

```

3406 \def\@Glsxtr@fullpl#1#2[#3]{%
3407   \glsdoifexists{#2}%
3408   {%
3409     \glssetabbrvfmt{\glscategory{#2}}%
3410     \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
3411     \let\glsifplural\@firstoftwo
3412     \let\glscapscase\@secondofthree
3413     \let\glsinsert\@empty
3414     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
3415     \glsxtrsetupfulldefs
3416     \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3417   }%
3418   \glspostlinkhook
3419 }

```

\Glsxtrfullpl Plural full form (all upper case).

```

3420 \newrobustcmd*{\Glsxtrfullpl}{\@glshyp@opt\@ns@Glsxtrfullpl}
3421 \newcommand*\ns@Glsxtrfullpl[2][]{%
3422   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
3423   {\@Glsxtr@fullpl{#1}{#2}[]}%
3424 }

```

\@Glsxtr@fullpl Low-level macro:

```

3425 \def\@Glsxtr@fullpl#1#2[#3]{%
3426   \glsdoifexists{#2}%
3427   {%
3428     \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
3429     \let\glsifplural\@firstoftwo
3430     \let\glscapscase\@thirdofthree
3431     \let\glsinsert\@empty
3432     \def\glscustomtext{%
3433       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
3434     \glsxtrsetupfulldefs
3435     \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3436   }%
3437   \glspostlinkhook
3438 }

```

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

\glsxtrshort

```

3439 \newrobustcmd*{\glsxtrshort}{\@glshyp@opt\@ns@glsxtrshort}

```

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

```

3440 \newcommand*{\ns@glstrshort}[2] [] {%
3441   \new@ifnextchar[{\@glstrshort{#1}{#2}}{\@glstrshort{#1}{#2} [] }%
3442 }

```

Read in the final optional argument:

```

3443 \def\@glstrshort#1#2[#3] {%
3444   \glstoifexists{#2}%
3445   {%

```

Need to make sure \glsabbrvfont is set correctly.

```

3446     \glsssetabbrvfmt{\glscategory{#2}}%
3447     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3448     \let\glstrifwasfirstuse\@secondoftwo
3449     \let\gl@ifplural\@secondoftwo
3450     \let\glscapscase\@firstofthree
3451     \let\glinsert\@empty
3452     \def\glscustomtext{%
3453       \glsabbrvfont{\glsaccessshort{#2}}#3%
3454     }%
3455     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3456   }%
3457   \glspostlinkhook
3458 }

```

\Glsxtrshort

```

3459 \newrobustcmd*{\Glsxtrshort}{\@gl@hyp@opt\ns@Glsxtrshort}

```

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

```

3460 \newcommand*{\ns@Glsxtrshort}[2] [] {%
3461   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} [] }%
3462 }

```

Read in the final optional argument:

```

3463 \def\@Glsxtrshort#1#2[#3] {%
3464   \glstoifexists{#2}%
3465   {%
3466     \glsssetabbrvfmt{\glscategory{#2}}%
3467     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3468     \let\glstrifwasfirstuse\@secondoftwo
3469     \let\gl@ifplural\@secondoftwo
3470     \let\glscapscase\@secondofthree
3471     \let\glinsert\@empty
3472     \def\glscustomtext{%
3473       \Glsabbrvfont{\Glsaccessshort{#2}}#3%
3474     }%
3475     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3476   }%
3477   \glspostlinkhook
3478 }

```

\GLSxtrshort

```

3479 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\ns@GLSxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
3480 \newcommand*{\ns@GLSxtrshort}[2] [] {%
3481   \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} []}%
3482 }

    Read in the final optional argument:
3483 \def\@GLSxtrshort#1#2[#3] {%
3484   \glsdoifexists{#2}%
3485   {%
3486     \glssetabbrvfmt{\glscategory{#2}}%
3487     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3488     \let\glsxtrifwasfirstuse\@secondoftwo
3489     \let\glsifplural\@secondoftwo
3490     \let\glsapscase\@thirdofthree
3491     \let\glsinsert\@empty
3492     \def\glscustomtext{%
3493       \mfirstucMakeUppercase{\glsabbrvfont{\glsaccessshort{#2}}#3}%
3494     }%
3495     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3496   }%
3497   \glspostlinkhook
3498 }

```

\glsxtrlong

```

3499 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
3500 \newcommand*{\ns@glsxtrlong}[2] [] {%
3501   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2} []}%
3502 }

    Read in the final optional argument:
3503 \def\@glsxtrlong#1#2[#3] {%
3504   \glsdoifexists{#2}%
3505   {%
3506     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3507     \let\glsxtrifwasfirstuse\@secondoftwo
3508     \let\glsifplural\@secondoftwo
3509     \let\glsapscase\@firstofthree
3510     \let\glsinsert\@empty
3511     \def\glscustomtext{\glsaccesslong{#2}#3}%
3512     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3513   }%
3514   \glspostlinkhook
3515 }

```

\Glsxtrlong

```

3516 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\ns@Glsxtrlong}

```

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

```
3517 \newcommand*{\ns@Glsxtrlong}[2] [] {%
3518   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
3519 }
```

Read in the final optional argument:

```
3520 \def\@Glsxtrlong#1#2[#3] {%
3521   \glsdoifexists{#2}%
3522   {%
3523     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3524     \let\glxtrifwasfirstuse\@secondoftwo
3525     \let\gl@ifplural\@secondoftwo
3526     \let\glscapscase\@secondofthree
3527     \let\glinsert\@empty
3528     \def\glscustomtext{\Glsaccesslong{#2}#3}%
3529     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3530   }%
3531   \glspostlinkhook
3532 }
```

\GLSxtrlong

```
3533 \newrobustcmd*{\GLSxtrlong}{\@gl@hyp@opt\ns@GLSxtrlong}
```

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

```
3534 \newcommand*{\ns@GLSxtrlong}[2] [] {%
3535   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
3536 }
```

Read in the final optional argument:

```
3537 \def\@GLSxtrlong#1#2[#3] {%
3538   \glsdoifexists{#2}%
3539   {%
3540     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3541     \let\glxtrifwasfirstuse\@secondoftwo
3542     \let\gl@ifplural\@secondoftwo
3543     \let\glscapscase\@thirdofthree
3544     \let\glinsert\@empty
3545     \def\glscustomtext{\mfirstucMakeUppercase{\Glsaccesslong{#2}#3}}%
3546     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3547   }%
3548   \glspostlinkhook
3549 }
```

Plural short forms:

\glxtrshortpl

```
3550 \newrobustcmd*{\glxtrshortpl}{\@gl@hyp@opt\ns@glxtrshortpl}
```

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

```
3551 \newcommand*{\ns@glxtrshortpl}[2] [] {%
3552   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2} []}%
3553 }
```

Read in the final optional argument:

```

3554 \def\@glxtrshortpl#1#2[#3]{%
3555   \glstoifexists{#2}%
3556   {%
3557     \glsetabbrvfmt{\glscategory{#2}}%
3558     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3559     \let\glxtrifwasfirstuse\@secondoftwo
3560     \let\gl@ifplural\@firstoftwo
3561     \let\glscapscase\@firstofthree
3562     \let\glinsert\@empty
3563     \def\glscustomtext{%
3564       \glabbrvfont{\Glsaccessshortpl{#2}}#3%
3565     }%
3566     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3567   }%
3568   \glspostlinkhook
3569 }

```

\Glsxtrshortpl

```

3570 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\ns@Glsxtrshortpl}

```

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

```

3571 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
3572   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
3573 }

```

Read in the final optional argument:

```

3574 \def\@Glsxtrshortpl#1#2[#3]{%
3575   \glstoifexists{#2}%
3576   {%
3577     \glsetabbrvfmt{\glscategory{#2}}%
3578     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3579     \let\glxtrifwasfirstuse\@secondoftwo
3580     \let\gl@ifplural\@firstoftwo
3581     \let\glscapscase\@secondofthree
3582     \let\glinsert\@empty
3583     \def\glscustomtext{%
3584       \Glsabbrvfont{\Glsaccessshortpl{#2}}#3%
3585     }%
3586     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3587   }%
3588   \glspostlinkhook
3589 }

```

\GLSxtrshortpl

```

3590 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\ns@GLSxtrshortpl}

```

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

```

3591 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
3592   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
3593 }

```

Read in the final optional argument:

```
3594 \def\@GLSxtrshortpl#1#2[#3]{%
3595   \glsdoifexists{#2}%
3596   {%
3597     \glsetabbrvfmt{\glscategory{#2}}%
3598     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3599     \let\glxtrifwasfirstuse\@secondoftwo
3600     \let\glsifplural\@firstoftwo
3601     \let\glscapscase\@thirdofthree
3602     \let\glsinsert\@empty
3603     \def\glscustomtext{%
3604       \mfirstucMakeUppercase{\glsabbrvfont{\glsaccessshortpl{#2}}#3}%
3605     }%
3606     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3607   }%
3608   \glspostlinkhook
3609 }
```

Plural long forms:

`\glxtrlongpl`

```
3610 \newrobustcmd*{\glxtrlongpl}{\@gl@hyp@opt\@ns@glxtrlongpl}
```

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

```
3611 \newcommand*{\ns@glxtrlongpl}[2][ ]{%
3612   \new@ifnextchar[{\@glxtrlongpl{#1}{#2}}{\@glxtrlongpl{#1}{#2}[]}%
3613 }
```

Read in the final optional argument:

```
3614 \def\@glxtrlongpl#1#2[#3]{%
3615   \glsdoifexists{#2}%
3616   {%
3617     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3618     \let\glxtrifwasfirstuse\@secondoftwo
3619     \let\glsifplural\@firstoftwo
3620     \let\glscapscase\@firstofthree
3621     \let\glsinsert\@empty
3622     \def\glscustomtext{\glsaccesslongpl{#2}#3}%
3623     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3624   }%
3625   \glspostlinkhook
3626 }
```

`\Glsxtrlongpl`

```
3627 \newrobustcmd*{\Glsxtrlongpl}{\@gl@hyp@opt\@ns@Glsxtrlongpl}
```

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

```
3628 \newcommand*{\ns@Glsxtrlongpl}[2][ ]{%
3629   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2}[]}%
3630 }
```

Read in the final optional argument:

```
3631 \def\@GLSxtrlongpl#1#2[#3]{%
3632   \glsdoifexists{#2}%
3633   {%
3634     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3635     \let\glxtrifwasfirstuse\@secondoftwo
3636     \let\glsifplural\@firstoftwo
3637     \let\glscapscase\@secondofthree
3638     \let\glsinsert\@empty
3639     \def\glscustomtext{\Glsaccesslongpl{#2}#3}%
3640     \@gl@link[#1]{#2}{\csname gls\@glstype @entryfmt\endcsname}%
3641   }%
3642   \glspostlinkhook
3643 }
```

\GLSxtrlongpl

```
3644 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\@ns@GLSxtrlongpl}
```

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

```
3645 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
3646   \new@ifnextchar[\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
3647 }
```

Read in the final optional argument:

```
3648 \def\@GLSxtrlongpl#1#2[#3]{%
3649   \glsdoifexists{#2}%
3650   {%
3651     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3652     \let\glxtrifwasfirstuse\@secondoftwo
3653     \let\glsifplural\@firstoftwo
3654     \let\glscapscase\@thirdofthree
3655     \let\glsinsert\@empty
3656     \def\glscustomtext{\mfirstucMakeUppercase{\Glsaccesslongpl{#2}#3}}%
3657     \@gl@link[#1]{#2}{\csname gls\@glstype @entryfmt\endcsname}%
3658   }%
3659   \glspostlinkhook
3660 }
```

\glssetabbrvfmt Set the current format for the given category (or the abbreviation category if unset).

```
3661 \newcommand*{\glssetabbrvfmt}[1]{%
3662   \ifcsdef{@glsabbrv@current@#1}%
3663   {\glxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
3664   {\glxtr@applyabbrvfmt{@glsabbrv@current@abbreviation}}%
3665 }
```

sxtrgenabbrvfmt Similar to \glsgenacfmt, but for abbreviations.

```
3666 \newcommand*{\glxsxtrgenabbrvfmt}{%
3667   \ifdefempty\glscustomtext
3668   {%
```

3669 \ifglsused\glslabel
 3670 {%

Subsequent use:

3671 \glsifplural
 3672 {%

Subsequent plural form:

3673 \glscapscase
 3674 {%

Subsequent plural form, don't adjust case:

3675 \glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert
 3676 }%
 3677 {%

Subsequent plural form, make first letter upper case:

3678 \glsabbrvfont{\Glsaccessshortpl{\glslabel}}\glsinsert
 3679 }%
 3680 {%

Subsequent plural form, all caps:

3681 \mfirstucMakeUppercase
 3682 {\glsabbrvfont{\glsaccessshortpl{\glslabel}}\glsinsert}%
 3683 }%
 3684 }%
 3685 {%

Subsequent singular form

3686 \glscapscase
 3687 {%

Subsequent singular form, don't adjust case:

3688 \glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert
 3689 }%
 3690 {%

Subsequent singular form, make first letter upper case:

3691 \glsabbrvfont{\Glsaccessshort{\glslabel}}\glsinsert
 3692 }%
 3693 {%

Subsequent singular form, all caps:

3694 \mfirstucMakeUppercase
 3695 {\glsabbrvfont{\glsaccessshort{\glslabel}}\glsinsert}%
 3696 }%
 3697 }%
 3698 }%
 3699 {%

First use:

3700 \glsifplural
 3701 {%

First use plural form:

```
3702      \glscapscase
3703      {%
```

First use plural form, don't adjust case:

```
3704      \glxtrfullplformat{\glslabel}{\glsinsert}%
3705      }%
3706      {%
```

First use plural form, make first letter upper case:

```
3707      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
3708      }%
3709      {%
```

First use plural form, all caps:

```
3710      \mfirstucMakeUppercase
3711      {\glxtrfullplformat{\glslabel}{\glsinsert}}%
3712      }%
3713      }%
3714      {%
```

First use singular form

```
3715      \glscapscase
3716      {%
```

First use singular form, don't adjust case:

```
3717      \glxtrfullformat{\glslabel}{\glsinsert}%
3718      }%
3719      {%
```

First use singular form, make first letter upper case:

```
3720      \Glsxtrfullformat{\glslabel}{\glsinsert}%
3721      }%
3722      {%
```

First use singular form, all caps:

```
3723      \mfirstucMakeUppercase
3724      {\glxtrfullformat{\glslabel}{\glsinsert}}%
3725      }%
3726      }%
3727      }%
3728      }%
3729      {%
```

User supplied text.

```
3730      \glscustomtext
3731      }%
3732 }
```

1.6.1 Abbreviation Styles Setup

abbreviationstyle

```

3733 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
3734   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
3735   {%
3736     \PackageError{glossaries-extra}{Undefined abbreviation style '#2'}{}}%
3737   }%
3738   {%

```

Have abbreviations already been defined for this category?

```

3739   \ifcsstring{@glsabbrv@current@#1}{#2}%
3740   {%

```

Style already set.

```

3741   }%
3742   {%
3743     \def\@glsxtr@dostylewarn{%
3744       \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
3745       {%
3746         \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
3747           style has been switched \MessageBreak
3748           for category '#1', \MessageBreak
3749           but there have already been entries \MessageBreak
3750           defined for this category. Unwanted \MessageBreak
3751           side-effects may result}}}%
3752       \@endfortrue
3753     }%
3754     \@glsxtr@dostylewarn

```

Set up the style for the given category.

```

3755     \csdef{@glsabbrv@current@#1}{#2}%
3756     \glsxtr@applyabbrvstyle{#2}%
3757   }%
3758   }%
3759 }

```

applyabbrvstyle Apply the abbreviation style without existence check.

```

3760 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
3761   \csuse{@glsabbrv@dispstyle@setup@#1}%
3762   \csuse{@glsabbrv@dispstyle@fmts@#1}%
3763 }

```

r@applyabbrvfmt Only apply the style formats.

```

3764 \newcommand*{\glsxtr@applyabbrvfmt}[1]{%
3765   \csuse{@glsabbrv@dispstyle@fmts@#1}%
3766 }

```

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.

```

3767 \newcommand*{\newabbreviationstyle}[3]{%
3768   \ifcsdef{@glsabbrv@dispstyle@#1}
3769   {%
3770     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
3771       defined}{}}%
3772   }%
3773   {%
3774     \csdef{@glsabbrv@dispstyle@setup@#1}{%
      Initialise hook to do nothing. The style may change this.
3775       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
3776       #2}%
3777     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
      Assume in-line form is the same as first use. The style may change this.
3778       \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
3779       \renewcommand*{\glxtrinlinefullformat}{\Glsxtrfullformat}%
3780       \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
3781       \renewcommand*{\glxtrinlinefullplformat}{\Glsxtrfullplformat}%
3782       #3}%
3783     }%
3784 }

```

eAbbrStyleSetup

```

3785 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
3786   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
3787   {%
3788     \PackageError{glossaries-extra}%
3789     {Unknown abbreviation style definitions ‘#1’}{}}%
3790   }%
3791   {%
3792     \csname @glsabbrv@dispstyle@setup@#1\endcsname
3793   }%
3794 }

```

seAbbrStyleFmt

```

3795 \newcommand*{\GlsXtrUseAbbrStyleFmt}[1]{%
3796   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
3797   {%
3798     \PackageError{glossaries-extra}%
3799     {Unknown abbreviation style formats ‘#1’}{}}%
3800   }%
3801   {%
3802     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
3803   }%
3804 }

```

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.

`\glstrinsertinside` Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```
3805 \newif\ifglstrinsertinside
3806 \glstrinsertinsidefalse
```

long-short

```
3807 \newabbreviationstyle{long-short}%
3808 {%
3809   \renewcommand*{\CustomAbbreviationFields}{%
3810     name={\protect\glsabbrvfont{\the\glsshorttok}},
3811     sort={\the\glsshorttok},
3812     first={\protect\glsfirstlongfont{\the\glslongtok}%
3813       \protect\glstrfullsep{\the\glslabeltok}%
3814       (\protect\glsfirstabbrvfont{\the\glsshorttok})},%
3815     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
3816       \protect\glstrfullsep{\the\glslabeltok}%
3817       (\protect\glsfirstabbrvfont{\the\glsshortpltok})},%
3818     plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
3819     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
3820   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3821     \glshasattribute{\the\glslabeltok}{regular}%
3822     {%
3823       \glssetattribute{\the\glslabeltok}{regular}{false}%
3824     }%
3825   }%
3826 }%
3827 }%
3828 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
3829   \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
3830   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
3831   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
3832   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
3833   \renewcommand*{\glstrfullformat}[2]{%
3834     \glsfirstlongfont{\glsaccesslong{##1}\ifglstrinsertinside##2\fi}%
```

```

3835 \ifglxtrinsertinside\else##2\fi
3836 \glxtrfullsep{##1}%
3837 (\glsfirstabbrvfont{\glssaccessshort{##1}})%
3838 }%
3839 \renewcommand*{\glxtrfullplformat}[2]{%
3840 \glsfirstlongfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
3841 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
3842 (\glsfirstabbrvfont{\glssaccessshortpl{##1}})%
3843 }%
3844 \renewcommand*{\Glsxtrfullformat}[2]{%
3845 \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
3846 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
3847 (\glsfirstabbrvfont{\glssaccessshort{##1}})%
3848 }%
3849 \renewcommand*{\Glsxtrfullplformat}[2]{%
3850 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
3851 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
3852 (\glsfirstabbrvfont{\glssaccessshortpl{##1}})%
3853 }%
3854 }

```

Set this as the default style for general abbreviations:

```

3855 \setabbreviationstyle{long-short}

```

long-short-desc User supplies description. The long form is included in the name.

```

3856 \newabbreviationstyle{long-short-desc}%
3857 {%
3858 \renewcommand*{\CustomAbbreviationFields}{%
3859 name={\protect\glxtrfullformat{\the\glslabeltok}{}},
3860 sort={\the\glssshorttok},%
3861 first={\protect\glsfirstlongfont{\the\glslongtok}%
3862 \protect\glxtrfullsep{\the\glslabeltok}%
3863 (\protect\glsfirstabbrvfont{\the\glssshorttok})},%
3864 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
3865 \protect\glxtrfullsep{\the\glslabeltok}%
3866 (\protect\glsfirstabbrvfont{\the\glssshortpltok})},%
3867 plural={\protect\glsabbvfont{\the\glssshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

3868 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3869 \glshasattribute{\the\glslabeltok}{regular}%
3870 {%
3871 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
3872 }%
3873 }%
3874 }%
3875 }%
3876 {%
3877 \GlsXtrUseAbbrStyleFmts{long-short}%
3878 }

```

short-long Short form followed by long form in parenthesis on first use.

```
3879 \newabbreviationstyle{short-long}%
3880 {%
3881   \renewcommand*{\CustomAbbreviationFields}{%
3882     name={\protect\glsabbrvfont{\the\glsshorttok}},
3883     sort={\the\glsshorttok},
3884     description={\the\glslongtok},%
3885     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
3886     \protect\glsxtrfullsep{\the\glslabeltok}%
3887     (\protect\glsfirstlongfont{\the\glslongtok}}),%
3888     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
3889     \protect\glsxtrfullsep{\the\glslabeltok}%
3890     (\protect\glsfirstlongfont{\the\glslongpltok}}),%
3891     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
3892 }
```

Unset the regular attribute if it has been set.

```
3892 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3893   \glshasattribute{\the\glslabeltok}{regular}%
3894   {%
3895     \glissetattribute{\the\glslabeltok}{regular}{false}%
3896   }%
3897 }%
3898 }%
3899 }%
3900 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
3901 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
3902 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
3903 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
3904 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
3905 \renewcommand*{\glsxtrfullformat}[2]{%
3906   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
3907   \ifglsxtrininsertinside\else##2\fi
3908   \glsxtrfullsep{##1}%
3909   (\glsfirstlongfont{\glsaccesslong{##1}})%
3910 }%
3911 \renewcommand*{\glsxtrfullplformat}[2]{%
3912   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
3913   \ifglsxtrininsertinside\else##2\fi
3914   \glsxtrfullsep{##1}%
3915   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
3916 }%
3917 \renewcommand*{\Glsxtrfullformat}[2]{%
3918   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
3919   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
3920   (\glsfirstlongfont{\Glsaccesslong{##1}})%
3921 }%
```

```

3922 \renewcommand*{\Glsxtrfullplformat}[2]{%
3923   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
3924   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
3925   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
3926 }%
3927 }

```

short-long-desc User supplies description. The long form is included in the name.

```

3928 \newabbreviationstyle{short-long-desc}%
3929 {%
3930   \renewcommand*{\CustomAbbreviationFields}{%
3931     name={\protect\glxtrfullformat{\the\glslabeltok}{}},
3932     sort={\the\glsshorttok},%
3933     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
3934       \protect\glxtrfullsep{\the\glslabeltok}%
3935       (\protect\glsfirstlongfont{\the\glslongtok})},%
3936     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
3937       \protect\glxtrfullsep{\the\glslabeltok}%
3938       (\protect\glsfirstlongfont{\the\glslongpltok})},%
3939     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
3940   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3941     \glshasattribute{\the\glslabeltok}{regular}%
3942     {%
3943       \glissetattribute{\the\glslabeltok}{regular}{false}%
3944     }%
3945     {}%
3946   }%
3947 }%
3948 {%
3949   \GlsXtrUseAbbrStyleFmts{short-long}%
3950 }

```

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?)

```

3951 \newabbreviationstyle{footnote}%
3952 {%
3953   \renewcommand*{\CustomAbbreviationFields}{%
3954     name={\protect\glsabbrvfont{\the\glsshorttok}},
3955     sort={\the\glsshorttok},
3956     description={\the\glslongtok},%
3957     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
3958       \protect\footnote{\protect\glsfirstlongfont{\the\glslongtok}}},%
3959     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
3960       \protect\footnote{\protect\glsfirstlongfont{\the\glslongpltok}}},%
3961     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.

```

3962 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
3963   \glsssetattribute{\the\glslabelltok}{nohyperfirst}{true}%
3964   \glshasattribute{\the\glslabelltok}{regular}%
3965   {%
3966     \glsssetattribute{\the\glslabelltok}{regular}{false}%
3967   }%
3968   {}%
3969 }%
3970 }%
3971 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

3972 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
3973 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
3974 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
3975 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

3976 \renewcommand*{\glsxtrfullformat}[2]{%
3977   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
3978   \ifglsxtrininsertinside\else##2\fi
3979   \protect\footnote{\glsfirstlongfont{\glsaccesslong{##1}}}%
3980 }%
3981 \renewcommand*{\glsxtrfullplformat}[2]{%
3982   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
3983   \ifglsxtrininsertinside\else##2\fi
3984   \protect\footnote{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
3985 }%
3986 \renewcommand*{\Glsxtrfullformat}[2]{%
3987   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
3988   \ifglsxtrininsertinside\else##2\fi
3989   \protect\footnote{\glsfirstlongfont{\glsaccesslong{##1}}}%
3990 }%
3991 \renewcommand*{\Glsxtrfullplformat}[2]{%
3992   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
3993   \ifglsxtrininsertinside\else##2\fi
3994   \protect\footnote{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
3995 }%

```

The first use full form and the inline full form use the short (long) style.

```

3996 \renewcommand*{\glsxtrinlinefullformat}[2]{%
3997   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
3998   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
3999   (\glsfirstlongfont{\glsaccesslong{##1}})%
4000 }%
4001 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
4002   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
4003   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%

```



```

4004 (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4005 }%
4006 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4007   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4008   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4009   (\glsfirstlongfont{\glsaccesslong{##1}})%
4010 }%
4011 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4012   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4013   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4014   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4015 }%
4016 }

```

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`.

```

4017 \newabbreviationstyle{postfootnote}%
4018 {%
4019   \renewcommand*{\CustomAbbreviationFields}{%
4020     name={\protect\glsabbrvfont{\the\glsshorttok}},
4021     sort={\the\glsshorttok},
4022     description={\the\glslongtok},%
4023     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
4024       \protect\footnote{\protect\glsfirstlongfont{\the\glslongtok}}},%
4025     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
4026       \protect\footnote{\protect\glsfirstlongfont{\the\glslongpltok}}},%
4027     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

4028 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4029   \csdef{glxtrpostlink\glscategorylabel}{%
4030     \glxtrifwasfirstuse
4031     {%
4032       \glxtrdopostpunc{\protect\footnote
4033         {\glsfirstlongfont{\glsentrylong{\glslabel}}}}%
4034     }%
4035   }%
4036 }%
4037 \glshasattribute{\the\glslabeltok}{regular}%
4038 {%
4039   \glissetattribute{\the\glslabeltok}{regular}{false}%
4040 }%
4041 {}%
4042 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

4043 \renewcommand*{\glxtrsetupfulldefs}{%

```

```

4044 \let\glxtrifwasfirstuse\@secondoftwo
4045 }%
4046 }%
4047 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4048 \renewcommand*\abbrvpluralsuffix{\glspluralsuffix}%
4049 \renewcommand*\glabbrvfont[1]{\glabbrvdefaultfont{##1}}%
4050 \renewcommand*\glfirstabbrvfont[1]{\glfirstabbrvdefaultfont{##1}}%
4051 \renewcommand*\glfirstlongfont[1]{\glfirstlongdefaultfont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

4052 \renewcommand*\glxtrfullformat[2]{%
4053   \glfirstabbrvfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
4054   \ifglxtrininsertinside\else##2\fi
4055 }%
4056 \renewcommand*\glxtrfullplformat[2]{%
4057   \glfirstabbrvfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
4058   \ifglxtrininsertinside\else##2\fi
4059 }%
4060 \renewcommand*\Glsxtrfullformat[2]{%
4061   \glfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
4062   \ifglxtrininsertinside\else##2\fi
4063 }%
4064 \renewcommand*\Glsxtrfullplformat[2]{%
4065   \glfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
4066   \ifglxtrininsertinside\else##2\fi
4067 }%

```

The first use full form and the inline full form use the short (long) style.

```

4068 \renewcommand*\glxtrininlinefullformat[2]{%
4069   \glfirstabbrvfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
4070   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
4071   (\glfirstlongfont{\glaccesslong{##1}})%
4072 }%
4073 \renewcommand*\glxtrininlinefullplformat[2]{%
4074   \glfirstabbrvfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
4075   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
4076   (\glfirstlongfont{\glaccesslongpl{##1}})%
4077 }%
4078 \renewcommand*\Glsxtrininlinefullformat[2]{%
4079   \glfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
4080   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
4081   (\glfirstlongfont{\Glsaccesslong{##1}})%
4082 }%
4083 \renewcommand*\Glsxtrininlinefullplformat[2]{%
4084   \glfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
4085   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
4086   (\glfirstlongfont{\Glsaccesslongpl{##1}})%
4087 }%
4088 }

```

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.

```

4089 \newabbreviationstyle{short}%
4090 {%
4091   \renewcommand*{\CustomAbbreviationFields}{%
4092     name={\protect\glsabbrvfont{\the\glsshorttok}},
4093     sort={\the\glsshorttok},
4094     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
4095     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
4096     text={\protect\glsabbrvfont{\the\glsshorttok}},
4097     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
4098     description={\the\glslongtok}}%
4099   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4100     \glssetattribute{\the\glslabeltok}{regular}{true}}%
4101 }%
4102 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4103 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4104 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4105 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4106 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

4107 \renewcommand*{\glsxtrinlinefullformat}[2]{%
4108   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
4109   \ifglsxtrininsertinside##2\fi}%
4110   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4111   (\glsfirstlongfont{\glsaccesslong{##1}})%
4112 }%
4113 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
4114   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
4115   \ifglsxtrininsertinside##2\fi}%
4116   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4117   (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4118 }%
4119 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4120   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
4121   \ifglsxtrininsertinside##2\fi}%
4122   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4123   (\glsfirstlongfont{\Glsaccesslong{##1}})%
4124 }%
4125 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4126   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
4127   \ifglsxtrininsertinside##2\fi}%
4128   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
4129   (\glsfirstlongfont{\Glsaccesslongpl{##1}})%
4130 }%

```

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.

```

4131 \renewcommand*{\glxtrfullformat}[2]{%
4132   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4133   \ifglxtrinsertinside\else##2\fi
4134 }%
4135 \renewcommand*{\glxtrfullplformat}[2]{%
4136   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4137   \ifglxtrinsertinside\else##2\fi
4138 }%
4139 \renewcommand*{\Glsxtrfullformat}[2]{%
4140   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4141   \ifglxtrinsertinside\else##2\fi
4142 }%
4143 \renewcommand*{\Glsxtrfullplformat}[2]{%
4144   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4145   \ifglxtrinsertinside\else##2\fi
4146 }%
4147 }

```

Set this as the default style for acronyms:

```

4148 \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.

```

4149 \newabbreviationstyle{short-desc}%
4150 {%
4151   \renewcommand*{\CustomAbbreviationFields}{%
4152     name={\protect\glxtrinlinefullformat{\the\glslabeltok}{}},
4153     sort={\the\glsshorttok},
4154     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
4155     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
4156     text={\protect\glsabbrvfont{\the\glsshorttok}},
4157     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
4158     description={\the\glslongtok}}%
4159   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4160     \glssetattribute{\the\glslabeltok}{regular}{true}}%
4161 }%
4162 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4163 \renewcommand*{\abbrvpluralsuffix}{\glspluralsuffix}%
4164 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
4165 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
4166 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

4167 \renewcommand*{\glxtrinlinefullformat}[2]{%
4168   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%

```

```

4169     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4170     (\glsfirstlongfont{\glsaccesslong{##1}})%
4171 }%
4172 \renewcommand*{\glxtrinlinefullplformat}[2]{%
4173     \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4174     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4175     (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4176 }%
4177 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
4178     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4179     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4180     (\glsfirstlongfont{\glsaccesslong{##1}})%
4181 }%
4182 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
4183     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4184     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
4185     (\glsfirstlongfont{\glsaccesslongpl{##1}})%
4186 }%

```

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.

```

4187 \renewcommand*{\glxtrfullformat}[2]{%
4188     \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4189     \ifglxtrinsertinside\else##2\fi
4190 }%
4191 \renewcommand*{\glxtrfullplformat}[2]{%
4192     \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4193     \ifglxtrinsertinside\else##2\fi
4194 }%
4195 \renewcommand*{\Glsxtrfullformat}[2]{%
4196     \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
4197     \ifglxtrinsertinside\else##2\fi
4198 }%
4199 \renewcommand*{\Glsxtrfullplformat}[2]{%
4200     \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
4201     \ifglxtrinsertinside\else##2\fi
4202 }%
4203 }

```

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.

```

4204 \newabbreviationstyle{long-desc}%
4205 {%
4206     \renewcommand*{\CustomAbbreviationFields}{%
4207         name={\protect\protect\glsfirstlongfont{\the\glslongtok}},
4208         sort={\the\glslongtok},
4209         first={\protect\glsfirstlongfont{\the\glslongtok}},
4210         firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},

```

```

4211     text={\the\glslongtok},
4212     plural={\the\glslongpltok}%
4213 }%
4214 \renewcommand*\GlsXtrPostNewAbbreviation{%
4215     \glssetattribute{\the\glslabeltok}{regular}{true}}%
4216 }%
4217 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

4218 \renewcommand*\abbrvpluralsuffix{\glspluralsuffix}%
4219 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
4220 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
4221 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%

```

The inline full form displays the long format followed by the short form in parentheses.

```

4222 \renewcommand*\glsxtrinlinefullformat[2]{%
4223     \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
4224     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4225     (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
4226 }%
4227 \renewcommand*\glsxtrinlinefullplformat[2]{%
4228     \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
4229     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4230     (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4231 }%
4232 \renewcommand*\Glsxtrinlinefullformat[2]{%
4233     \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
4234     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4235     (\protect\glsfirstabbrvfont{\glsaccessshort{##1}})%
4236 }%
4237 \renewcommand*\Glsxtrinlinefullplformat[2]{%
4238     \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
4239     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
4240     (\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}})%
4241 }%

```

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.

```

4242 \renewcommand*\glsxtrfullformat[2]{%
4243     \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
4244     \ifglsxtrinsertinside\else##2\fi
4245 }%
4246 \renewcommand*\glsxtrfullplformat[2]{%
4247     \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
4248     \ifglsxtrinsertinside\else##2\fi
4249 }%
4250 \renewcommand*\Glsxtrfullformat[2]{%
4251     \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
4252     \ifglsxtrinsertinside\else##2\fi
4253 }%
4254 \renewcommand*\Glsxtrfullplformat[2]{%

```

```

4255 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
4256 \ifglxtrinsertinside\else##2\fi
4257 }%
4258 }

```

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.

```

4259 \newabbreviationstyle{long}%
4260 {%
4261 \renewcommand*{\CustomAbbreviationFields}{%
4262 name={\protect\glsabbrvfont{\the\glsshorttok}},
4263 sort={\the\glsshorttok},
4264 first={\protect\glsfirstlongfont{\the\glslongtok}},
4265 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
4266 text={\the\glslongtok},
4267 plural={\the\glslongpltok},%
4268 description={\the\glslongtok}%
4269 }%
4270 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
4271 \glssetattribute{\the\glslabeltok}{regular}{true}}%
4272 }%
4273 {%
4274 \GlsXtrUseAbbrStyleFmts{long-desc}%
4275 }

```

1.6.3 Predefined Styles (Small Capitals)

These styles use:

`\glxtrscfont`

```

4276 \newcommand*{\glxtrscfont}[1]{\textsc{#1}}

```

and for the default short form suffix:

`\glxtrscsuffix`

```

4277 \newcommand*{\glxtrscsuffix}{\glstextup{\glspluralsuffix}}

```

`long-short-sc`

```

4278 \newabbreviationstyle{long-short-sc}%
4279 {%
4280 \GlsXtrUseAbbrStyleSetup{long-short}%
4281 }%
4282 {%

```

Mostly as long-short style:

```

4283 \GlsXtrUseAbbrStyleFmts{long-short}%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4284 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4285 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
4286 }
```

g-short-sc-desc

```
4287 \newabbreviationstyle{long-short-sc-desc}%
4288 {%
4289 \GlsXtrUseAbbrStyleSetup{long-short-desc}%
4290 }%
4291 {%
```

Mostly as long-short-desc style:

```
4292 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4293 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4294 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
4295 }
```

Now the short (long) version

```
4296 \newabbreviationstyle{short-sc-long}%
4297 {%
4298 \GlsXtrUseAbbrStyleSetup{short-long}%
4299 }%
4300 {%
```

Mostly as short-long style:

```
4301 \GlsXtrUseAbbrStyleFmts{short-long}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4302 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4303 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
4304 }
```

As before but user provides description

```
4305 \newabbreviationstyle{short-sc-long-desc}%
4306 {%
4307 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
4308 }%
4309 {%
```

Mostly as short-long-desc style:

```
4310 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4311 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
4312 \renewcommand*{\glssabrvfont[1]{\glxtrscfont{##1}}}%
4313 }
```


short-sc

```
4314 \newabbreviationstyle{short-sc}%  
4315 {%  
4316   \GlsXtrUseAbbrStyleSetup{short}%  
4317 }%  
4318 {%
```

Mostly as short style:

```
4319   \GlsXtrUseAbbrStyleFmts{short}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4320   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%  
4321   \renewcommand*{\glsabbrvfont[1]}{\glstrscfont{##1}}%  
4322 }
```

short-sc-desc

```
4323 \newabbreviationstyle{short-sc-desc}%  
4324 {%  
4325   \GlsXtrUseAbbrStyleSetup{short-desc}%  
4326 }%  
4327 {%
```

Mostly as short style:

```
4328   \GlsXtrUseAbbrStyleFmts{short-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4329   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%  
4330   \renewcommand*{\glsabbrvfont[1]}{\glstrscfont{##1}}%  
4331 }
```

long-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
4332 \newabbreviationstyle{long-sc}%  
4333 {%  
4334   \GlsXtrUseAbbrStyleSetup{long}%  
4335 }%  
4336 {%
```

Mostly as long style:

```
4337   \GlsXtrUseAbbrStyleFmts{long}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4338   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%  
4339   \renewcommand*{\glsabbrvfont[1]}{\glstrscfont{##1}}%  
4340 }
```

long-desc-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
4341 \newabbreviationstyle{long-desc-sc}%  
4342 {%  
4343   \GlsXtrUseAbbrStyleSetup{long-desc}%
```

```
4344 }%
4345 {%
```

Mostly as long style:

```
4346 \GlsXtrUseAbbrStyleFmts{long-desc}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4347 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
4348 \renewcommand*{\glsabbrvfont}[1]{\glstrscfont{##1}}%
4349 }
```

footnote-sc

```
4350 \newabbreviationstyle{footnote-sc}%
4351 {%
4352 \GlsXtrUseAbbrStyleSetup{footnote}%
4353 }%
4354 {%
```

Mostly as long style:

```
4355 \GlsXtrUseAbbrStyleFmts{footnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4356 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
4357 \renewcommand*{\glsabbrvfont}[1]{\glstrscfont{##1}}%
4358 }
```

postfootnote-sc

```
4359 \newabbreviationstyle{postfootnote-sc}%
4360 {%
4361 \GlsXtrUseAbbrStyleSetup{postfootnote}%
4362 }%
4363 {%
```

Mostly as long style:

```
4364 \GlsXtrUseAbbrStyleFmts{postfootnote}%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
4365 \renewcommand*{\abbrvpluralsuffix}{\protect\glstrscsuffix}%
4366 \renewcommand*{\glsabbrvfont}[1]{\glstrscfont{##1}}%
4367 }
```

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:

`\glstrsmfont`

```
4368 \newcommand*{\glstrsmfont}[1]{\textsmaller{##1}}
```

and for the default short form suffix:

\glxtrmsuffix

```
4369 \newcommand*{\glxtrmsuffix}{\glspluralsuffix}
```

long-short-sm

```
4370 \newabbreviationstyle{long-short-sm}%
4371 {%
4372   \GlsXtrUseAbbrStyleSetup{long-short}%
4373 }%
4374 {%
```

Mostly as long-short style:

```
4375   \GlsXtrUseAbbrStyleFmts{long-short}%
4376   \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
4377   \renewcommand*\abbrvpluralsuffix{\protect\glxtrmsuffix}%
4378 }
```

g-short-sm-desc

```
4379 \newabbreviationstyle{long-short-sm-desc}%
4380 {%
4381   \GlsXtrUseAbbrStyleSetup{long-short-desc}%
4382 }%
4383 {%
```

Mostly as long-short-desc style:

```
4384   \GlsXtrUseAbbrStyleFmts{long-short-desc}%
4385   \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
4386   \renewcommand*\abbrvpluralsuffix{\protect\glxtrmsuffix}%
4387 }
```

short-sm-long Now the short (long) version

```
4388 \newabbreviationstyle{short-sm-long}%
4389 {%
4390   \GlsXtrUseAbbrStyleSetup{short-long}%
4391 }%
4392 {%
```

Mostly as short-long style:

```
4393   \GlsXtrUseAbbrStyleFmts{short-long}%
4394   \renewcommand*\glsabbrvfont[1]{\glxtrsmfont{##1}}%
4395   \renewcommand*\abbrvpluralsuffix{\protect\glxtrmsuffix}%
4396 }
```

rt-sm-long-desc As before but user provides description

```
4397 \newabbreviationstyle{short-sm-long-desc}%
4398 {%
4399   \GlsXtrUseAbbrStyleSetup{short-long-desc}%
4400 }%
4401 {%
```

Mostly as short-long-desc style:

```
4402 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
4403 \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
4404 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4405 }
```

short-sm

```
4406 \newabbreviationstyle{short-sm}%
4407 {%
4408 \GlsXtrUseAbbrStyleSetup{short}%
4409 }%
4410 {%
```

Mostly as short style:

```
4411 \GlsXtrUseAbbrStyleFmts{short}%
4412 \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
4413 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4414 }
```

short-sm-desc

```
4415 \newabbreviationstyle{short-sm-desc}%
4416 {%
4417 \GlsXtrUseAbbrStyleSetup{short-desc}%
4418 }%
4419 {%
```

Mostly as short style:

```
4420 \GlsXtrUseAbbrStyleFmts{short-desc}%
4421 \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
4422 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4423 }
```

long-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
4424 \newabbreviationstyle{long-sm}%
4425 {%
4426 \GlsXtrUseAbbrStyleSetup{long}%
4427 }%
4428 {%
```

Mostly as long style:

```
4429 \GlsXtrUseAbbrStyleFmts{long}%
4430 \renewcommand*\glsabbrvfont[1]{\glstrsmfont{##1}}%
4431 \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4432 }
```

long-desc-sm The smaller font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
4433 \newabbreviationstyle{long-desc-sm}%
```

```

4434 {%
4435   \GlsXtrUseAbbrStyleSetup{long-desc}%
4436 }%
4437 {%

```

Mostly as long style:

```

4438   \GlsXtrUseAbbrStyleFmts{long-desc}%
4439   \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%
4440   \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4441 }

```

footnote-sm

```

4442 \newabbreviationstyle{footnote-sm}%
4443 {%
4444   \GlsXtrUseAbbrStyleSetup{footnote}%
4445 }%
4446 {%

```

Mostly as long style:

```

4447   \GlsXtrUseAbbrStyleFmts{footnote}%
4448   \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%
4449   \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4450 }

```

postfootnote-sm

```

4451 \newabbreviationstyle{postfootnote-sm}%
4452 {%
4453   \GlsXtrUseAbbrStyleSetup{postfootnote}%
4454 }%
4455 {%

```

Mostly as long style:

```

4456   \GlsXtrUseAbbrStyleFmts{postfootnote}%
4457   \renewcommand*\glsabrvfont[1]{\glstrsmfont{##1}}%
4458   \renewcommand*\abbrvpluralsuffix{\protect\glstrsmsuffix}%
4459 }

```

1.6.5 Predefined Styles (Emphasized)

These styles use \emph for the short form.

long-short-em

```

4460 \newabbreviationstyle{long-short-em}%
4461 {%
4462   \GlsXtrUseAbbrStyleSetup{long-short}%
4463 }%
4464 {%

```

Mostly as long-short style:

```

4465   \GlsXtrUseAbbrStyleFmts{long-short}%

```

```

4466 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4467 }

```

g-short-em-desc

```

4468 \newabbreviationstyle{long-short-em-desc}%
4469 {%
4470 \GlsXtrUseAbbrStyleSetup{long-short-desc}%
4471 }%
4472 {%
    Mostly as long-short-desc style:
4473 \GlsXtrUseAbbrStyleFmts{long-short-desc}%
4474 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4475 }

```

short-em-long Now the short (long) version

```

4476 \newabbreviationstyle{short-em-long}%
4477 {%
4478 \GlsXtrUseAbbrStyleSetup{short-long}%
4479 }%
4480 {%
    Mostly as short-long style:
4481 \GlsXtrUseAbbrStyleFmts{short-long}%
4482 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4483 }

```

rt-em-long-desc As before but user provides description

```

4484 \newabbreviationstyle{short-em-long-desc}%
4485 {%
4486 \GlsXtrUseAbbrStyleSetup{short-long-desc}%
4487 }%
4488 {%
    Mostly as short-long-desc style:
4489 \GlsXtrUseAbbrStyleFmts{short-long-desc}%
4490 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4491 }

```

short-em

```

4492 \newabbreviationstyle{short-em}%
4493 {%
4494 \GlsXtrUseAbbrStyleSetup{short}%
4495 }%
4496 {%
    Mostly as short style:
4497 \GlsXtrUseAbbrStyleFmts{short}%
4498 \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4499 }

```

short-em-desc

```
4500 \newabbreviationstyle{short-em-desc}%
4501 {%
4502   \GlsXtrUseAbbrStyleSetup{short-desc}%
4503 }%
4504 {%
  Mostly as short style:
4505   \GlsXtrUseAbbrStyleFmts{short-desc}%
4506   \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4507 }
```

long-em The emphasized font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
4508 \newabbreviationstyle{long-em}%
4509 {%
4510   \GlsXtrUseAbbrStyleSetup{long}%
4511 }%
4512 {%
  Mostly as long style:
4513   \GlsXtrUseAbbrStyleFmts{long}%
4514   \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4515 }
```

long-desc-em The emphasized font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
4516 \newabbreviationstyle{long-desc-em}%
4517 {%
4518   \GlsXtrUseAbbrStyleSetup{long-desc}%
4519 }%
4520 {%
  Mostly as long style:
4521   \GlsXtrUseAbbrStyleFmts{long-desc}%
4522   \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4523 }
```

footnote-em

```
4524 \newabbreviationstyle{footnote-em}%
4525 {%
4526   \GlsXtrUseAbbrStyleSetup{footnote}%
4527 }%
4528 {%
  Mostly as long style:
4529   \GlsXtrUseAbbrStyleFmts{footnote}%
4530   \renewcommand*\glsabbrvfont[1]{\emph{##1}}%
4531 }
```

postfootnote-em

```
4532 \newabbreviationstyle{postfootnote-em}%  
4533 {%  
4534   \GlsXtrUseAbbrStyleSetup{postfootnote}%  
4535 }%  
4536 {%
```

Mostly as long style:

```
4537   \GlsXtrUseAbbrStyleFmts{postfootnote}%  
4538   \renewcommand*\glsabbrvfont[1]{\emph{##1}}%  
4539 }
```

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:

```
4540 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
4541 \renewcommand*{\markright}[1]{%  
4542   \glsxtrmarkhook  
4543   \@glsxtr@org@markright{#1}%  
4544   \glsxtrrestoremarkhook  
4545 }
```


\markboth Save original definition:

```
4546 \let\@glxstr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
4547 \renewcommand*{\markboth}[2]{%
4548   \glxstrmarkhook
4549   \@glxstr@org@markboth{#1}{#2}%
4550   \glxstrrestoremarkhook
4551 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```
4552 \newcommand*{\glxstrRevertMarks}{%
4553   \let\markright\@glxstr@org@markright
4554   \let\markboth\@glxstr@org@markboth
4555 }
```

\glxstrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```
4556 \newcommand*{\glxstrmarkhook}{%
```

Save current definitions:

```
4557 \let\@glxstr@org@MakeUppercase\MakeUppercase
4558 \let\@glxstr@org@glxstrtitleshort\glxstrtitleshort
4559 \let\@glxstr@org@glxstrtitleshortpl\glxstrtitleshortpl
4560 \let\@glxstr@org@Glsxtrtitleshort\Glsxtrtitleshort
4561 \let\@glxstr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
4562 \let\@glxstr@org@glxstrtitletext\glxstrtitletext
4563 \let\@glxstr@org@Glsxtrtitletext\Glsxtrtitletext
4564 \let\@glxstr@org@glxstrtitleplural\glxstrtitleplural
4565 \let\@glxstr@org@Glsxtrtitleplural\Glsxtrtitleplural
4566 \let\@glxstr@org@glxstrtitlefirst\glxstrtitlefirst
4567 \let\@glxstr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
4568 \let\@glxstr@org@glxstrtitlefirstplural\glxstrtitlefirstplural
4569 \let\@glxstr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
4570 \let\@glxstr@org@glxstrtitlelong\glxstrtitlelong
4571 \let\@glxstr@org@glxstrtitlelongpl\glxstrtitlelongpl
4572 \let\@glxstr@org@Glsxtrtitlelong\Glsxtrtitlelong
4573 \let\@glxstr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
4574 \let\@glxstr@org@glxstrtitlefull\glxstrtitlefull
4575 \let\@glxstr@org@glxstrtitlefullpl\glxstrtitlefullpl
4576 \let\@glxstr@org@Glsxtrtitlefull\Glsxtrtitlefull
4577 \let\@glxstr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl
```

New definitions

```
4578 \let\MakeUppercase\MakeTextUppercase
4579 \let\glxstrtitleshort\glxstrheadshort
4580 \let\glxstrtitleshortpl\glxstrheadshortpl
4581 \let\Glsxtrtitleshort\Glsxtrheadshort
```

```

4582 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
4583 \let\glxstrtitletext\glxstrheadtext
4584 \let\Glsxtrtitletext\Glsxtrheadtext
4585 \let\glxstrtitleplural\glxstrheadplural
4586 \let\Glsxtrtitleplural\Glsxtrheadplural
4587 \let\glxstrtitlefirst\glxstrheadfirst
4588 \let\Glsxtrtitlefirst\Glsxtrheadfirst
4589 \let\glxstrtitlefirstplural\glxstrheadfirstplural
4590 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
4591 \let\glxstrtitlelong\glxstrheadlong
4592 \let\glxstrtitlelongpl\glxstrheadlongpl
4593 \let\Glsxtrtitlelong\Glsxtrheadlong
4594 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
4595 \let\glxstrtitlefull\glxstrheadfull
4596 \let\glxstrtitlefullpl\glxstrheadfullpl
4597 \let\Glsxtrtitlefull\Glsxtrheadfull
4598 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
4599 }

```

`\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.)

```

4600 \newcommand*{\glxstrrestoremarkhook}{%
4601   \let\MakeUppercase\@glxstr@org@MakeUppercase
4602   \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
4603   \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
4604   \let\Glsxtrtitleshort\@glxstr@org@Glsxtrtitleshort
4605   \let\Glsxtrtitleshortpl\@glxstr@org@Glsxtrtitleshortpl
4606   \let\glxstrtitletext\@glxstr@org@glxstrtitletext
4607   \let\Glsxtrtitletext\@glxstr@org@Glsxtrtitletext
4608   \let\glxstrtitleplural\@glxstr@org@glxstrtitleplural
4609   \let\Glsxtrtitleplural\@glxstr@org@Glsxtrtitleplural
4610   \let\glxstrtitlefirst\@glxstr@org@glxstrtitlefirst
4611   \let\Glsxtrtitlefirst\@glxstr@org@Glsxtrtitlefirst
4612   \let\glxstrtitlefirstplural\@glxstr@org@glxstrtitlefirstplural
4613   \let\Glsxtrtitlefirstplural\@glxstr@org@Glsxtrtitlefirstplural
4614   \let\glxstrtitlelong\@glxstr@org@glxstrtitlelong
4615   \let\glxstrtitlelongpl\@glxstr@org@glxstrtitlelongpl
4616   \let\Glsxtrtitlelong\@glxstr@org@Glsxtrtitlelong
4617   \let\Glsxtrtitlelongpl\@glxstr@org@Glsxtrtitlelongpl
4618   \let\glxstrtitlefull\@glxstr@org@glxstrtitlefull
4619   \let\glxstrtitlefullpl\@glxstr@org@glxstrtitlefullpl
4620   \let\Glsxtrtitlefull\@glxstr@org@Glsxtrtitlefull
4621   \let\Glsxtrtitlefullpl\@glxstr@org@Glsxtrtitlefullpl
4622 }

```

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.

```

4623 \newcommand*{\glsxtrheadshort}[1]{%
4624   \protect\NoCaseChange
4625   {%
4626     \glsifattribute{#1}{headuc}{true}%
4627     {%
4628       \GLSxtrshort[noindex,hyper=false]{#1}[]%
4629     }%
4630     {%
4631       \glsxtrshort[noindex,hyper=false]{#1}[]%
4632     }%
4633   }%
4634 }
```

`lsxtrtitleshort` Command to display short form of abbreviation in section title and table of contents.

```

4635 \newrobustcmd*{\glsxtrtitleshort}[1]{%
4636   \glsxtrshort[noindex,hyper=false]{#1}[]%
4637 }
```

`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.

```

4638 \newcommand*{\glsxtrheadshortpl}[1]{%
4639   \protect\NoCaseChange
4640   {%
4641     \glsifattribute{#1}{headuc}{true}%
4642     {%
4643       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
4644     }%
4645     {%
4646       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
4647     }%
4648   }%
4649 }
```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents.

```

4650 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
4651   \glsxtrshort[noindex,hyper=false]{#1}[]%
4652 }
```

`Glsxtrheadshort` Command used to display short form in the page header with the first letter converted to upper case.

```

4653 \newcommand*{\Glsxtrheadshort}[1]{%
4654   \protect\NoCaseChange
4655   {%
4656     \glsifattribute{#1}{headuc}{true}%
4657     {%
4658       \GLSxtrshort[noindex,hyper=false]{#1}[]%

```

```

4659 }%
4660 {%
4661     \Glsxtrshort[noindex,hyper=false]{#1}[]%
4662 }%
4663 }%
4664 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

4665 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
4666     \Glsxtrshort[noindex,hyper=false]{#1}[]%
4667 }

```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```

4668 \newcommand*{\Glsxtrheadshortpl}[1]{%
4669     \protect\NoCaseChange
4670 {%
4671     \glsifattribute{#1}{headuc}{true}%
4672     {%
4673         \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
4674     }%
4675     {%
4676         \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
4677     }%
4678 }%
4679 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

4680 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
4681     \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
4682 }

```

\glsxtrheadtext As above but for the text value.

```

4683 \newcommand*{\glsxtrheadtext}[1]{%
4684     \protect\NoCaseChange
4685 {%
4686     \glsifattribute{#1}{headuc}{true}%
4687     {%
4688         \GLStext[noindex,hyper=false]{#1}[]%
4689     }%
4690     {%
4691         \glstext[noindex,hyper=false]{#1}[]%
4692     }%
4693 }%
4694 }

```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```
4695 \newrobustcmd*{\glsxtrtitletext}[1]{%
4696   \glstext[noindex,hyper=false]{#1}[]%
4697 }
```

`\Glsxtrheadtext` First letter converted to upper case

```
4698 \newcommand*{\Glsxtrheadtext}[1]{%
4699   \protect\NoCaseChange
4700   {%
4701     \glsifattribute{#1}{headuc}{true}%
4702     {%
4703       \GLStext[noindex,hyper=false]{#1}[]%
4704     }%
4705     {%
4706       \Glstext[noindex,hyper=false]{#1}[]%
4707     }%
4708   }%
4709 }
```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```
4710 \newrobustcmd*{\Glsxtrtitletext}[1]{%
4711   \Glstext[noindex,hyper=false]{#1}[]%
4712 }
```

`lsxtrheadplural` As above but for the plural value.

```
4713 \newcommand*{\glsxtrheadplural}[1]{%
4714   \protect\NoCaseChange
4715   {%
4716     \glsifattribute{#1}{headuc}{true}%
4717     {%
4718       \GLSplural[noindex,hyper=false]{#1}[]%
4719     }%
4720     {%
4721       \glsplural[noindex,hyper=false]{#1}[]%
4722     }%
4723   }%
4724 }
```

`sxtrtitleplural` Command to display plural value in section title and table of contents.

```
4725 \newrobustcmd*{\glsxtrtitleplural}[1]{%
4726   \glsplural[noindex,hyper=false]{#1}[]%
4727 }
```

`lsxtrheadplural` Convert first letter to upper case.

```
4728 \newcommand*{\Glsxtrheadplural}[1]{%
4729   \protect\NoCaseChange
4730   {%
```

```

4731 \glsifattribute{#1}{headuc}{true}%
4732 {%
4733   \GLSplural[noindex,hyper=false]{#1}[]%
4734 }%
4735 {%
4736   \Glsplural[noindex,hyper=false]{#1}[]%
4737 }%
4738 }%
4739 }

```

gsxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```

4740 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
4741   \Glsplural[noindex,hyper=false]{#1}[]%
4742 }

```

glsxtrheadfirst As above but for the first value.

```

4743 \newcommand*{\glsxtrheadfirst}[1]{%
4744   \protect\NoCaseChange
4745   {%
4746     \glsifattribute{#1}{headuc}{true}%
4747     {%
4748       \GLSfirst[noindex,hyper=false]{#1}[]%
4749     }%
4750     {%
4751       \glsfirst[noindex,hyper=false]{#1}[]%
4752     }%
4753   }%
4754 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```

4755 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
4756   \glsfirst[noindex,hyper=false]{#1}[]%
4757 }

```

Glsxtrheadfirst First letter converted to upper case

```

4758 \newcommand*{\Glsxtrheadfirst}[1]{%
4759   \protect\NoCaseChange
4760   {%
4761     \glsifattribute{#1}{headuc}{true}%
4762     {%
4763       \GLSfirst[noindex,hyper=false]{#1}[]%
4764     }%
4765     {%
4766       \Glsfirst[noindex,hyper=false]{#1}[]%
4767     }%
4768   }%
4769 }

```

`lsxtrtitlefirst` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
4770 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
4771   \Glsfirst[noindex,hyper=false]{#1}[]%
4772 }
```

`headfirstplural` As above but for the firstplural value.

```
4773 \newcommand*{\Glsxtrheadfirstplural}[1]{%
4774   \protect\NoCaseChange
4775   {%
4776     \glsifattribute{#1}{headuc}{true}%
4777     {%
4778       \GLSfirstplural[noindex,hyper=false]{#1}[]%
4779     }%
4780     {%
4781       \glsfirstplural[noindex,hyper=false]{#1}[]%
4782     }%
4783   }%
4784 }
```

`itlefirstplural` Command to display firstplural value in section title and table of contents.

```
4785 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
4786   \glsfirstplural[noindex,hyper=false]{#1}[]%
4787 }
```

`headfirstplural` First letter converted to upper case

```
4788 \newcommand*{\Glsxtrheadfirstplural}[1]{%
4789   \protect\NoCaseChange
4790   {%
4791     \glsifattribute{#1}{headuc}{true}%
4792     {%
4793       \GLSfirstplural[noindex,hyper=false]{#1}[]%
4794     }%
4795     {%
4796       \Glsfirstplural[noindex,hyper=false]{#1}[]%
4797     }%
4798   }%
4799 }
```

`itlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
4800 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
4801   \Glsfirstplural[noindex,hyper=false]{#1}[]%
4802 }
```

`\Glsxtrheadlong` Command used to display long form in the page header.

```
4803 \newcommand*{\Glsxtrheadlong}[1]{%
4804   \protect\NoCaseChange
```

```

4805 {%
4806   \glsifattribute{#1}{headuc}{true}%
4807   {%
4808     \GLSxtrlong[noindex,hyper=false]{#1}[]%
4809   }%
4810   {%
4811     \glsxtrlong[noindex,hyper=false]{#1}[]%
4812   }%
4813 }%
4814 }

```

`\glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```

4815 \newrobustcmd*{\glsxtrtitlelong}[1]{%
4816   \glsxtrlong[noindex,hyper=false]{#1}[]%
4817 }

```

`\glsxtrheadlongpl` Command used to display plural long form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrlongpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

4818 \newcommand*{\glsxtrheadlongpl}[1]{%
4819   \protect\NoCaseChange
4820   {%
4821     \glsifattribute{#1}{headuc}{true}%
4822     {%
4823       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
4824     }%
4825     {%
4826       \glsxtrlongpl[noindex,hyper=false]{#1}[]%
4827     }%
4828   }%
4829 }

```

`\glsxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```

4830 \newrobustcmd*{\glsxtrtitlelongpl}[1]{%
4831   \glsxtrlong[noindex,hyper=false]{#1}[]%
4832 }

```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

4833 \newcommand*{\Glsxtrheadlong}[1]{%
4834   \protect\NoCaseChange
4835   {%
4836     \glsifattribute{#1}{headuc}{true}%
4837     {%
4838       \GLSxtrlong[noindex,hyper=false]{#1}[]%
4839     }%
4840     {%
4841       \Glsxtrlong[noindex,hyper=false]{#1}[]%

```



```

4842 }%
4843 }%
4844 }

```

Glsxtrtitlelong Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

4845 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
4846   \Glsxtrlong[noindex,hyper=false]{#1}[]%
4847 }

```

lsxtrheadlongpl Command used to display plural long form in the page header with the first letter converted to upper case.

```

4848 \newcommand*{\Glsxtrheadlongpl}[1]{%
4849   \protect\NoCaseChange
4850   {%
4851     \glsifattribute{#1}{headuc}{true}%
4852     {%
4853       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
4854     }%
4855     {%
4856       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
4857     }%
4858   }%
4859 }

```

sxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

4860 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
4861   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
4862 }

```

\glsxtrheadfull Command used to display full form in the page header.

```

4863 \newcommand*{\glsxtrheadfull}[1]{%
4864   \protect\NoCaseChange
4865   {%
4866     \glsifattribute{#1}{headuc}{true}%
4867     {%
4868       \GLSxtrfull[noindex,hyper=false]{#1}[]%
4869     }%
4870     {%
4871       \glsxtrfull[noindex,hyper=false]{#1}[]%
4872     }%
4873   }%
4874 }

```

glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents.

```

4875 \newrobustcmd*{\glsxtrtitlefull}[1]{%
4876   \glsxtrfull[noindex,hyper=false]{#1}[]%
4877 }

```

`lsxtrheadfullpl` Command used to display plural full form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrfullpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

4878 \newcommand*{\glxtrheadfullpl}[1]{%
4879   \protect\NoCaseChange
4880   {%
4881     \gl@ifattribute{#1}{headuc}{true}%
4882     {%
4883       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
4884     }%
4885     {%
4886       \glxtrfullpl[noindex,hyper=false]{#1}[]%
4887     }%
4888   }%
4889 }

```

`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents.

```

4890 \newrobustcmd*{\glxtrtitlefullpl}[1]{%
4891   \glxtrfullpl[noindex,hyper=false]{#1}[]%
4892 }

```

`\Glsxtrheadfull` Command used to display full form in the page header with the first letter converted to upper case.

```

4893 \newcommand*{\Glsxtrheadfull}[1]{%
4894   \protect\NoCaseChange
4895   {%
4896     \gl@ifattribute{#1}{headuc}{true}%
4897     {%
4898       \GLSxtrfull[noindex,hyper=false]{#1}[]%
4899     }%
4900     {%
4901       \Glsxtrfull[noindex,hyper=false]{#1}[]%
4902     }%
4903   }%
4904 }

```

`Glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

4905 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
4906   \Glsxtrfull[noindex,hyper=false]{#1}[]%
4907 }

```

`lsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```

4908 \newcommand*{\Glsxtrheadfullpl}[1]{%
4909   \protect\NoCaseChange
4910   {%
4911     \gl@ifattribute{#1}{headuc}{true}%

```

```

4912  {%
4913    \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
4914  }%
4915  {%
4916    \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
4917  }%
4918 }%
4919 }

```

`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

4920 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
4921   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
4922 }

```

`\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.

```

4923 \ifdef\texorpdfstring
4924 {
4925   \newcommand*{\glsfmtshort}[1]{%
4926     \texorpdfstring
4927       {\glsxtrtitleshort{#1}}%
4928       {\glsentryshort{#1}}%
4929   }
4930 }
4931 {
4932   \newcommand*{\glsfmtshort}[1]{%
4933     \glsxtrtitleshort{#1}}
4934 }

```

Similarly for the plural version.

```

\glsfmtshortpl
4935 \ifdef\texorpdfstring
4936 {
4937   \newcommand*{\glsfmtshortpl}[1]{%
4938     \texorpdfstring
4939       {\glsxtrtitleshortpl{#1}}%
4940       {\glsentryshortpl{#1}}%
4941   }
4942 }
4943 {
4944   \newcommand*{\glsfmtshortpl}[1]{%
4945     \glsxtrtitleshortpl{#1}}
4946 }

```

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).

```
4947 \ifdef\teorpdfstring
4948 {
4949   \newcommand*\Glsfmtshort}[1]{%
4950     \teorpdfstring
4951       {\Glsxtrtitleshort{#1}}%
4952       {\glsentryshort{#1}}%
4953   }
4954 }
4955 {
4956   \newcommand*\Glsfmtshort}[1]{%
4957     \Glsxtrtitleshort{#1}}
4958 }
```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```
4959 \ifdef\teorpdfstring
4960 {
4961   \newcommand*\Glsfmtshortpl}[1]{%
4962     \teorpdfstring
4963       {\Glsxtrtitleshortpl{#1}}%
4964       {\glsentryshortpl{#1}}%
4965   }
4966 }
4967 {
4968   \newcommand*\Glsfmtshortpl}[1]{%
4969     \Glsxtrtitleshortpl{#1}}
4970 }
```

`\glsfmttext` As above but for the text value.

```
4971 \ifdef\teorpdfstring
4972 {
4973   \newcommand*\glsfmttext}[1]{%
4974     \teorpdfstring
4975       {\glsxtrtitletext{#1}}%
4976       {\glsentrytext{#1}}%
4977   }
4978 }
4979 {
4980   \newcommand*\glsfmttext}[1]{%
4981     \glsxtrtitletext{#1}}
4982 }
```

`\Glsfmttext` First letter converted to upper case.

```
4983 \ifdef\teorpdfstring
4984 {
4985   \newcommand*\Glsfmttext}[1]{%
4986     \teorpdfstring
4987       {\Glsxtrtitletext{#1}}%
4988       {\glsentrytext{#1}}%
```

```

4989 }
4990 }
4991 {
4992   \newcommand*{\Glsfmttext}[1]{%
4993     \Glsxtrtitletext{#1}}
4994 }

```

`\glsfmtplural` As above but for the plural value.

```

4995 \ifdef\teorpdfstring
4996 {
4997   \newcommand*{\glsfmtplural}[1]{%
4998     \teorpdfstring
4999     {\Glsxtrtitleplural{#1}}%
5000     {\Glsentryplural{#1}}%
5001   }
5002 }
5003 {
5004   \newcommand*{\glsfmtplural}[1]{%
5005     \Glsxtrtitleplural{#1}}
5006 }

```

`\Glsfmtplural` First letter converted to upper case.

```

5007 \ifdef\teorpdfstring
5008 {
5009   \newcommand*{\Glsfmtplural}[1]{%
5010     \teorpdfstring
5011     {\Glsxtrtitleplural{#1}}%
5012     {\Glsentryplural{#1}}%
5013   }
5014 }
5015 {
5016   \newcommand*{\Glsfmtplural}[1]{%
5017     \Glsxtrtitleplural{#1}}
5018 }

```

`\glsfmtfirst` As above but for the first value.

```

5019 \ifdef\teorpdfstring
5020 {
5021   \newcommand*{\glsfmtfirst}[1]{%
5022     \teorpdfstring
5023     {\Glsxtrtitlefirst{#1}}%
5024     {\Glsentryfirst{#1}}%
5025   }
5026 }
5027 {
5028   \newcommand*{\glsfmtfirst}[1]{%
5029     \Glsxtrtitlefirst{#1}}
5030 }

```

`\Glsfmtfirst` First letter converted to upper case.

```
5031 \ifdef\txorpdfstring
5032 {
5033   \newcommand*\Glsfmtfirst}[1]{%
5034     \txorpdfstring
5035     {\Glsxtrtitlefirst{#1}}%
5036     {\glsentryfirst{#1}}%
5037   }
5038 }
5039 {
5040   \newcommand*\Glsfmtfirst}[1]{%
5041     \Glsxtrtitlefirst{#1}}
5042 }
```

`\glsfmtfirstpl` As above but for the firstplural value.

```
5043 \ifdef\txorpdfstring
5044 {
5045   \newcommand*\glsfmtfirstpl}[1]{%
5046     \txorpdfstring
5047     {\glsxtrtitlefirstplural{#1}}%
5048     {\glsentryfirstplural{#1}}%
5049   }
5050 }
5051 {
5052   \newcommand*\glsfmtfirstpl}[1]{%
5053     \glsxtrtitlefirstplural{#1}}
5054 }
```

`\Glsfmtfirstpl` First letter converted to upper case.

```
5055 \ifdef\txorpdfstring
5056 {
5057   \newcommand*\Glsfmtfirstpl}[1]{%
5058     \txorpdfstring
5059     {\Glsxtrtitlefirstplural{#1}}%
5060     {\glsentryfirstplural{#1}}%
5061   }
5062 }
5063 {
5064   \newcommand*\Glsfmtfirstpl}[1]{%
5065     \Glsxtrtitlefirstplural{#1}}
5066 }
```

`\glsfmtlong` As above but for the long value.

```
5067 \ifdef\txorpdfstring
5068 {
5069   \newcommand*\glsfmtlong}[1]{%
5070     \txorpdfstring
5071     {\glsxtrtitlelong{#1}}%
5072     {\glsentrylong{#1}}%
```

```

5073 }
5074 }
5075 {
5076   \newcommand*{\glsfmtlong}[1]{%
5077     \glsxtrtitlelong{#1}}
5078 }

```

`\Glsfmtlong` First letter converted to upper case.

```

5079 \ifdef\texorpdfstring
5080 {
5081   \newcommand*{\Glsfmtlong}[1]{%
5082     \texorpdfstring
5083       {\Glsxtrtitlelong{#1}}%
5084       {\glsentrylong{#1}}%
5085   }
5086 }
5087 {
5088   \newcommand*{\Glsfmtlong}[1]{%
5089     \Glsxtrtitlelong{#1}}
5090 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

5091 \ifdef\texorpdfstring
5092 {
5093   \newcommand*{\glsfmtlongpl}[1]{%
5094     \texorpdfstring
5095       {\glsxtrtitlelongpl{#1}}%
5096       {\glsentrylongpl{#1}}%
5097   }
5098 }
5099 {
5100   \newcommand*{\glsfmtlongpl}[1]{%
5101     \glsxtrtitlelongpl{#1}}
5102 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

5103 \ifdef\texorpdfstring
5104 {
5105   \newcommand*{\Glsfmtlongpl}[1]{%
5106     \texorpdfstring
5107       {\Glsxtrtitlelongpl{#1}}%
5108       {\glsentrylongpl{#1}}%
5109   }
5110 }
5111 {
5112   \newcommand*{\Glsfmtlongpl}[1]{%
5113     \Glsxtrtitlelongpl{#1}}
5114 }

```

`\glsfmtfull` In-line full format.

```
5115 \ifdef\txorpdfstring
5116 {
5117   \newcommand*\glsfmtfull}[1]{%
5118     \txorpdfstring
5119     {\glsxtrtitlefull{#1}}%
5120     {\glsxtrinlinefullformat{#1}-{}}%
5121   }
5122 }
5123 {
5124   \newcommand*\glsfmtfull}[1]{%
5125     \glsxtrtitlefull{#1}}
5126 }
```

`\Glsfmtfull` First letter converted to upper case.

```
5127 \ifdef\txorpdfstring
5128 {
5129   \newcommand*\Glsfmtfull}[1]{%
5130     \txorpdfstring
5131     {\Glsxtrtitlefull{#1}}%
5132     {\Glsxtrinlinefullformat{#1}-{}}%
5133   }
5134 }
5135 {
5136   \newcommand*\Glsfmtfull}[1]{%
5137     \Glsxtrtitlefull{#1}}
5138 }
```

`\glsfmtfullpl` In-line full plural format.

```
5139 \ifdef\txorpdfstring
5140 {
5141   \newcommand*\glsfmtfullpl}[1]{%
5142     \txorpdfstring
5143     {\glsxtrtitlefullpl{#1}}%
5144     {\glsxtrinlinefullplformat{#1}-{}}%
5145   }
5146 }
5147 {
5148   \newcommand*\glsfmtfullpl}[1]{%
5149     \glsxtrtitlefullpl{#1}}
5150 }
```

`\Glsfmtfullpl` First letter converted to upper case.

```
5151 \ifdef\txorpdfstring
5152 {
5153   \newcommand*\Glsfmtfullpl}[1]{%
5154     \txorpdfstring
5155     {\Glsxtrtitlefullpl{#1}}%
5156     {\Glsxtrinlinefullplformat{#1}-{}}%
5157   }
5158 }
```



```

5157 }
5158 }
5159 {
5160   \newcommand*{\Glsfmtfullpl}[1]{%
5161     \Glsxtrtitlefullpl{#1}}
5162 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

5163 \newcommand*{\RequireGlossariesExtraLang}[1]{%
5164   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
5165 }

```

`sariesExtraLang`

```

5166 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
5167   \ProvidesFile{glossariesxtr-#1.ldf}%
5168 }

```

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.)

```

5169 \@ifpackageloaded{tracklang}
5170 {%
5171   \AnyTrackedLanguages
5172   {%
5173     \ForEachTrackedDialect{\this@dialect}{%
5174       \IfTrackedLanguageFileExists{\this@dialect}%
5175       {glossariesxtr-}% prefix
5176       {.ldf}%
5177       {%
5178         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
5179       }%
5180     }%
5181   }%
5182 }%
5183 }%
5184 {}%
5185 }
5186 {}

```

Load `glossaries-extra-stylemods` if required.

```

5187 \@glsxtr@redefstyles

```

2 Style Adjustments (glossaries-extra-stylemods.sty)

This package adjusts the predefined styles so that they include the post description hook.

2.1 Package Initialisation

First identify package:

```
5188 \NeedsTeXFormat{LaTeX2e}
5189 \ProvidesPackage{glossaries-extra-stylemods}[2016/04/25 v1.02 (NLCT)]
```

Provide package options to automatically load required predefined styles. The simplest method is to just test for the existence of the file `glossary-⟨option⟩.sty`. Packages can't be loaded whilst the options are being processed, so save the list in `\@glxtr@loadstyles`.

`sxtr@loadstyles`

```
5190 \newcommand*{\@glxtr@loadstyles}{%
5191   \DeclareOption*{%
5192     \IfFileExists{glossary-\CurrentOption.sty}
5193       {\eappto\@glxtr@loadstyles{%
5194         \noexpand\RequirePackage{glossary-\CurrentOption}}}%
5195     {\PackageError{glossaries-extra-styles}%
5196       {Unknown option '\CurrentOption'}{}}
5197 }
```

Process the package options:

```
5198 \ProcessOptions
```

Load the required packages:

```
5199 \@glxtr@loadstyles
```

Adjust the styles that the post description hook added, but only for styles that have already been defined. All the tree styles in `glossary-tree` include the post description hook, so they don't require adjustment. Similarly for `glossary-mcols` which builds on the tree styles.

In case we have an old version of `glossaries`:

`ewglossarystyle`

```
5200 \providecommand{\renewglossarystyle}[2]{%
5201   \ifcsundef{@glstyle@#1}%
5202     {%
5203       \PackageError{glossaries}{Glossary style '#1' isn't already defined}{}%
5204     }%
```

```

5205 {%
5206   \csdef{@glsstyle@#1}{#2}%
5207 }%
5208 }

```

2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying.

```

5209 \ifcsdef{@glsstyle@listdotted}
5210 {%
5211   \renewglossarystyle{listdotted}{%
5212     \setglossarystyle{list}%
5213     \renewcommand*{\glossentry}[2]{%
5214       \item[]\makebox[\glslistdottedwidth][l]{%
5215         \glsentryitem{##1}%
5216         \glstarget{##1}{\glossentryname{##1}}%
5217         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
5218         \glossentrydesc{##1}\glspostdescription}%
5219     \renewcommand*{\subglossentry}[3]{%
5220       \item[]\makebox[\glslistdottedwidth][l]{%
5221         \glssubentryitem{##2}%
5222         \glstarget{##2}{\glossentryname{##2}}%
5223         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
5224         \glossentrydesc{##2}\glspostdescription}%
5225   }
5226 }
5227 {}

```

The sublistdotted style doesn't display the description for top-level entries. Sub-level entries use the listdottedstyle.

2.3 Longtable Styles

The three and four column styles require adjustment, but not the two column styles.

```

5228 \ifcsdef{@glsstyle@long3col}
5229 {%
5230   \renewglossarystyle{long3col}{%
5231     \renewenvironment{theglossary}%
5232       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
5233       {\end{longtable}}}%
5234     \renewcommand*{\glossaryheader}{}%
5235     \renewcommand*{\glsgroupheading}[1]{}%
5236     \renewcommand{\glossentry}[2]{%
5237       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
5238       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
5239     }%

```

```

5240 \renewcommand{\subglossentry}[3]{%
5241     &
5242     \glssubentryitem{##2}%
5243     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
5244     ##3\tabularnewline
5245 }%
5246 \renewcommand*{\glsgroupskip}{%
5247     \ifglsgnোগroupskip\else & &\tabularnewline\fi}%
5248 }
5249 }
5250 {}

```

Four column style:

```

5251 \ifcsdef{@glsstyle@long4col}
5252 {%
5253 \renewglossarystyle{long4col}{%
5254 \renewenvironment{theglossary}%
5255     {\begin{longtable}{llll}}%
5256     {\end{longtable}}%
5257 \renewcommand*{\glossaryheader}{}%
5258 \renewcommand*{\glsgroupheading}[1]{}%
5259 \renewcommand{\glossentry}[2]{%
5260     \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
5261     \glossentrydesc{##1}\glspostdescription &
5262     \glossentrysymbol{##1} &
5263     ##2\tabularnewline
5264 }%
5265 \renewcommand{\subglossentry}[3]{%
5266     &
5267     \glssubentryitem{##2}%
5268     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
5269     \glossentrysymbol{##2} & ##3\tabularnewline
5270 }%
5271 \renewcommand*{\glsgroupskip}{%
5272     \ifglsgnোগroupskip\else & & &\tabularnewline\fi}%
5273 }
5274 }
5275 {}

```

The styles in glossary-longbooktabs are all based on the styles in glossary-long, so no adjustments are needed for that package.

2.4 Long Ragged Styles

The three and four column styles require adjustment, but not the two column styles.

```

5276 \ifcsdef{@glsstyle@longragged3col}
5277 {%
5278 \renewglossarystyle{longragged3col}{%
5279 \renewenvironment{theglossary}%

```

```

5280     {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}%
5281       >{\raggedright}p{\glspagelistwidth}}}%
5282     {\end{longtable}}}%
5283     \renewcommand*{\glossaryheader}{}%
5284     \renewcommand*{\glsgroupheading}[1]{}%
5285     \renewcommand{\glossentry}[2]{%
5286       \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
5287       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
5288     }%
5289     \renewcommand{\subglossentry}[3]{%
5290       &
5291       \glssubentryitem{##2}%
5292       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
5293       ##3\tabularnewline
5294     }%
5295     \renewcommand*{\glsgroupskip}{%
5296       \ifglsgnoglobskip\else & &\tabularnewline\fi}%
5297   }
5298 }
5299 {}

```

Four column style:

```

5300 \ifcsdef{@glstyle@altlongragged4col}
5301 {%
5302   \renewglossarystyle{altlongragged4col}{%
5303     \renewenvironment{theglossary}%
5304       {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}l%
5305         >{\raggedright}p{\glspagelistwidth}}}%
5306       {\end{longtable}}}%
5307     \renewcommand*{\glossaryheader}{}%
5308     \renewcommand*{\glsgroupheading}[1]{}%
5309     \renewcommand{\glossentry}[2]{%
5310       \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
5311       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
5312       ##2\tabularnewline
5313     }%
5314     \renewcommand{\subglossentry}[3]{%
5315       &
5316       \glssubentryitem{##2}%
5317       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
5318       \glossentrysymbol{##2} & ##3\tabularnewline
5319     }%
5320     \renewcommand*{\glsgroupskip}{%
5321       \ifglsgnoglobskip\else & &\tabularnewline\fi}%
5322   }
5323 }
5324 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment, but not the two column styles.

```
5325 \ifcsdef{@glsstyle@super3col}
5326 {%
5327   \renewglossarystyle{super3col}{%
5328     \renewenvironment{theglossary}%
5329       {\tablehead{}\tabletail{}}%
5330       \begin{supertabular}{\lp{\glsdescwidth}p{\glspagelistwidth}}}%
5331     {\end{supertabular}}%
5332     \renewcommand*{\glossaryheader}{}%
5333     \renewcommand*{\glsgroupheading}[1]{}%
5334     \renewcommand{\glossentry}[2]{%
5335       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
5336       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
5337     }%
5338     \renewcommand{\subglossentry}[3]{%
5339       &
5340       \glssubentryitem{##2}%
5341       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
5342       ##3\tabularnewline
5343     }%
5344     \renewcommand*{\glsgroupskip}{%
5345       \ifglsgnোগroupskip\else & \tabularnewline\fi}%
5346   }
5347 }
5348 {}
```

Four column styles:

```
5349 \ifcsdef{@glsstyle@super4col}
5350 {%
5351   \renewglossarystyle{super4col}{%
5352     \renewenvironment{theglossary}%
5353       {\tablehead{}\tabletail{}}%
5354       \begin{supertabular}{\llll}}%
5355     {\end{supertabular}}%
5356     \renewcommand*{\glossaryheader}{}%
5357     \renewcommand*{\glsgroupheading}[1]{}%
5358     \renewcommand{\glossentry}[2]{%
5359       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
5360       \glossentrydesc{##1}\glspostdescription &
5361       \glossentrysymbol{##1} & ##2\tabularnewline
5362     }%
5363     \renewcommand{\subglossentry}[3]{%
5364       &
5365       \glssubentryitem{##2}%
5366       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
5367       \glossentrysymbol{##2} & ##3\tabularnewline
5368     }%
5369     \renewcommand*{\glsgroupskip}{%
```

```

5370      \ifglsnogroupskip\else & &\tabularnewline\fi}%
5371  }
5372 }
5373 {}

```

2.6 Super Ragged Styles

The three and four column styles require adjustment, but not the two column styles.

```

5374 \ifcsdef{@glsstyle@superragged3col}
5375 {%
5376   \renewglossarystyle{superragged3col}{%
5377     \renewenvironment{theglossary}%
5378       {\tablehead{}\tabletail}%
5379       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%
5380         >{\raggedright}p{\glspagelistwidth}}}%
5381       {\end{supertabular}}}%
5382   \renewcommand*{\glossaryheader}{}%
5383   \renewcommand*{\glsgroupheading}[1]{}%
5384   \renewcommand{\glossentry}[2]{%
5385     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
5386     \glossentrydesc{##1}\glspostdescription &
5387     ##2\tabularnewline
5388   }%
5389   \renewcommand{\subglossentry}[3]{%
5390     &
5391     \glssubentryitem{##2}%
5392     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
5393     ##3\tabularnewline
5394   }%
5395   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
5396     &\tabularnewline\fi}%
5397 }
5398 }
5399 {}

```

Four columns:

```

5400 \ifcsdef{@glsstyle@altsuperragged4col}
5401 {%
5402   \renewglossarystyle{altsuperragged4col}{%
5403     \renewenvironment{theglossary}%
5404       {\tablehead{}\tabletail}%
5405       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}1%
5406         >{\raggedright}p{\glspagelistwidth}}}%
5407       {\end{supertabular}}}%
5408   \renewcommand*{\glossaryheader}{}%
5409   \renewcommand{\glossentry}[2]{%
5410     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
5411     \glossentrydesc{##1}\glspostdescription &
5412     \glossentrysymbol{##1} & ##2\tabularnewline

```

```

5413 }%
5414 \renewcommand{\subglossentry}[3]{%
5415     &
5416     \glssubentryitem{##2}%
5417     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
5418     \glossentrysymbol{##2} & ##3\tabularnewline
5419 }%
5420 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & &
5421     &\tabularnewline\fi}%
5422 }
5423 }
5424 {}

```

2.7 Inline Style

The inline style is dealt with slightly differently. The `\glspostdescription` hook is actually in `\glspostinline`, which is called at the end of the glossary. The original definition of `\glspostinline` also includes a space, which is unnecessary. Here, instead of redefining the inline style, just redefine `\glspostinline` and `\glsinlinedescformat`.

```

5425 \ifdef{\@glsstyle@inline}
5426 {%
5427     \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
5428     Just use \glsxtrpostdescription instead of \glspostdescription.
5429     \renewcommand*{\glsinlinedescformat}[3]{%
5430         \space#1\glsxtrpostdescription}
5431     \renewcommand*{\glsinlinesubdescformat}[3]{%
5432         #1\glsxtrpostdescription}
5433 }
5434
5435 \Reset the default style
5436 \ifx\@glossary@default@style\relax
5437 \else
5438     \setglossarystyle{\@glxtr@current@style}
5439 \fi

```


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)		\@Glsymbolplural@: added redefini-	
General: Initial experimental release	4	tion	24
0.2 (2015-11-30)		\@Glstext@: added redefinition	21
\Glsfmtshort: new	148	\@Glsuseri@: added redefinition	25
\glsfmtshort: new	147	\@Glsuserii@: added redefinition	25
\Glsfmtshortpl: new	148	\@Glsuseriii@: added redefinition	25
\glsfmtshortpl: new	147	\@Glsuseriv@: added redefinition	25
short: switched inline full form to short		\@Glsuserv@: added redefinition	26
(long)	123	\@Glsuservi@: added redefinition	26
0.3 (2015-12-02)		\@acrlong: added redefinition	28
\@ACRlong: added redefinition	29	\@acrlongpl: added redefinition	29
\@ACRlongpl: added redefinition	30	\@acrshort: added redefinition	26
\@ACRshort: added redefinition	27	\@acrshortpl: added redefinition	27
\@ACRshortpl: added redefinition	28	\@gls@field@link: added optional ar-	
\@Acrlong: added redefinition	29	gument	21
\@Acrlongpl: added redefinition	30	\@glsdescplural@: added redefinition	23
\@Acrshort: added redefinition	27	\@glsfirst@: added redefinition	21
\@Acrshortpl: added redefinition	28	\@glsfirstplural@: added redefinition	22
\@GLSdesc@: added redefinition	23	\@glsplural: added redefinition	22
\@GLSdescplural@: added redefinition	24	\@glssymbolplural@: added redefini-	
\@GLSfirst@: added redefinition	22	tion	24
\@GLSname@: added redefinition	23	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural: added redefinition	22	new	63
\@GLSsymbol@: added redefinition	24	\@glsxtr@field@linkdefs: new	21
\@GLSsymbolplural@: added redefini-		\@glsxtr@insertdots: new	98
tion	25	\@print@glossary: added redefinition	60
\@GLStext@: added redefinition	21	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	25	\abbrvdefaultfont	102
\@GLSuserii@: added redefinition	25	\glsaccessdesc: new	70
\@GLSuseriii@: added redefinition	25	\glsaccessdescplural: new	71
\@GLSuseriv@: added redefinition	26	\glsaccessfirst: new	67
\@GLSuserv@: added redefinition	26	\glsaccessfirstplural: new	68
\@GLSuservi@: added redefinition	26	\Glsaccesslong: new	73
\@Glsdesc@: added redefinition	23	\glsaccesslong: new	73
\@Glsdescplural@: added redefinition	24	\glsaccessname: new	64
\@Glsfirst@: added redefinition	21	\glsaccessplural: new	66
\@Glsfirstplural@: added redefinition	23	\Glsaccessshort: new	72
\@Glsname@: added redefinition	23	\glsaccessshort: new	72
\@Glsplural: added redefinition	22	\Glsaccessshortpl: new	73
\@Glsymbol@: added redefinition	24	\glsaccessshortpl: new	73

\glsaccesssymbol: new	69	\cGLSpl@: new	46
\glsaccesssymbolplural: new	70	\@glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	65	new	41
\glstentryfmt: added check for short ..	20	\cGLS: new	45
\glslongpltok: new	98	\cGLSformat: new	46
\glsshortpltok: new	98	\cGLSpl: new	46
\glsxtrdiscardperiod: added check		\cGLSplformat: new	46
for plural	95	\GlossariesExtraWarningNoLine:	
\GLSxtrlongpl: new	111	new	6
\Glsxtrlongpl: new	110	\glsenableentrycount: new	41
\glsxtrlongpl: new	110	\glsfirstabbrvdefaultfont: new ..	102
\glsxtrNoGlossaryWarning: new	9	\glsfirstlongdefaultfont: new ...	102
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirst: new	150
new	95	\glsfmtfirst: new	149
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtfirstpl: new	150
new	95	\glsfmtfirstpl: new	150
\glsxtrpostlinkendsentence: new ..	95	\glsfmtplural: new	149
\GLSxtrshortpl: new	109	\glsfmtplural: new	149
\Glsxtrshortpl: new	109	\Glsfmtshort: changed to use	
\glsxtrshortpl: new	108	\Glsxtrtitleshort	148
short-long-desc: fixed name to use		renamed from \Glstentryfmtshort .	148
\glslabeltok	119	\glsfmtshort: changed to use	
\newabbreviation: fixed family name in		\glsxtrtitleshort	147
\setkeys	99	renamed from \glstentryfmtshort .	147
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	117	\Glsxtrtitleshortpl	148
0.4 (2015-12-03)		renamed from \Glstentryfmtshortpl	148
\@glsxtr@doabbreviationsdef: added		\glsfmtshortpl: changed to use	
redefinition of \acronymtype	6	\glsxtrtitleshortpl	147
\Glsfmtshort: changed to use		renamed from \glstentryfmtshortpl	147
\Glsxtrshort	148	\Glsfmttext: new	148
\glsfmtshort: changed to use		\glsfmttext: new	148
\glsxtrshort	147	\glshasattribute: new	79
\Glsfmtshortpl: changed to use		\glshascategoryattribute: new	78
\glsxtrshortpl	148	\GlsXtrEnableEntryCounting: new ..	40
\glsfmtshortpl: changed to use		\glsxtrifcounttrigger: new	43
\glsxtrshortpl	147	\glsxtrscfont: new	127
\glsxtrifemptyglossary: new	11	\glsxtrscsuffix: new	127
\glsxtrnewnumber: added extra argu-		\glsxtrsmfont: new	130
ment	82	\glsxtrsmsuffix: new	131
\glsxtrnewsymbol: added extra argu-		short-em: new	134
ment	81	short-em-desc: new	135
\MakeAcronymsAbbreviations: set the		short-em-long: new	134
default type to \acronymtype	54	short-em-long-desc: new	134
\newterm: fixed name argument	81	short-sm: new	132
0.5 (2015-12-07)		short-sm-desc: new	132
\cGLS: new	45	short-sm-long: new	131
\cGLS@: new	46	short-sm-long-desc: new	131
\cGLSpl: new	46	long-desc-em: new	135

long-desc-sm: new	132	\glxtrheadtext: now uses headuc at-tribute	140
long-em: new	135	short-long: switch off regular attribute if set	118
long-short-em: new	133	short-long-desc: switch off regular attribute if set	119
long-short-em-desc: new	134	long-short: switch off regular attribute if set	116
long-short-sm: new	131	long-short-desc: switch off regular attribute if set	117
long-short-sm-desc: new	131	footnote: switch off regular attribute if set	120
long-sm: new	132	postfootnote: switch off regular attribute if set	121
footnote-em: new	135		
footnote-sc: new	130	0.5.2 (2015-12-08)	
footnote-sm: new	133	\@GLSdesc@: added	71
postfootnote-em: new	136	\@GLSdescplural@: added	72
postfootnote-sc: new	130	\@GLSfirst@: added	68
postfootnote-sm: new	133	\@GLSfirstplural@: added	69
0.5.1 (2015-12-02)		\@GLSname@: added	65
\Glsaccessstext: new	65	\@GLSplural@: added	67
0.5.1 (2015-12-07)		\@GLSsymbol@: added	69
\@glxtr@doaccsupp: new	9	\@GLSsymbolplural@: added	70
General: removed \ifglxtruseuhead	138	\@GLStext@: added	66
\Glsaccessdesc: new	71	\@GLSdesc@: added	71
\Glsaccessdescplural: new	72	\@GLSdescplural@: added	72
\Glsaccessfirst: new	67	\@GLSfirst@: added	67
\Glsaccessfirstplural: new	68	\@GLSfirstplural@: added	68
\Glsaccessname: new	64	\@GLSname@: added	65
\Glsaccessplural: new	66	\@GLsplural@: added	66
\Glsaccesssymbol: new	69	\@GLsymbol@: added	69
\Glsaccesssymbolplural: new	70	\@GLsymbolplural@: added	70
\Glsxtrheadfirst: now uses headuc attribute	142	\@GLstext@: added	66
\glxtrheadfirst: now uses headuc attribute	142	\@glSdesc@: added	71
\Glsxtrheadfirstplural: now uses headuc attribute	143	\@glSdescplural@: added	72
\glxtrheadfirstplural: now uses headuc attribute	143	\@glSfirst@: added	67
\Glsxtrheadplural: now uses headuc attribute	141	\@glSfirstplural@: added	68
\glxtrheadplural: now uses headuc attribute	141	\@glSname@: added	64
\Glsxtrheadshort: now uses headuc attribute	139	\@glSplural@: added	66
\glxtrheadshort: now uses headuc attribute	139	\@glSsymbol@: added	69
\Glsxtrheadshortpl: now uses headuc attribute	140	\@glSsymbolplural@: added	70
\glxtrheadshortpl: now uses headuc attribute	139	\@glStext@: added	65
\Glsxtrheadtext: now uses headuc attribute	141	\@glxtr@activate@initialtagging: new	93
		\@glxtr@do@titlecaps@warn: new ..	93
		\@glxtr@tag: new	93
		General: fixed typo in glossaries-accsupp and tidied up code to use just one \@ifpackageloaded	64

removed \glsxtrabbrvfmt	111	0.5.4 ()	
\glossaryentrynumbers:added	19	\@glsxtr@setentryunitcountunsetattr:	
\Glossentrydesc:added	91	new	52
\Glossentryname:added	86	0.5.4 (2015-12-15)	
\Glossentrysymbol:added	91	\@newglossaryentry@defunitcounters:	
\glossentrysymbol:added	91	new	47
\GLSaccessdesc:new	71, 76	\@GLSxtr@p@acrlong@:new	39
\GLSaccessdescplural:new	72, 76	\@GLSxtr@p@acrlongpl@:new	39
\GLSaccessfirst:new	67, 75	\@GLSxtr@p@acrshort@:new	39
\GLSaccessfirstplural:new	68, 75	\@GLSxtr@p@acrshortpl@:new	39
\GLSaccesslong:new	74, 77	\@GLSxtr@p@long@:new	38
\GLSaccesslongpl:new	74, 77	\@GLSxtr@p@longpl@:new	39
\Glsaccesslongpl:new	74	\@GLSxtr@p@plural@:new	37
\glsaccesslongpl:new	74	\@GLSxtr@p@short@:new	38
\GLSaccessname:new	65, 74	\@GLSxtr@p@shortpl@:new	38
\GLSaccessplural:new	67, 75	\@GLSxtr@p@text@:new	37
\GLSaccessshort:new	73, 77	\@GlsXtrEnableOnTheFly:new	15
\GLSaccessshortpl:new	73, 77	\@Glsxtr:new	16
\GLSaccesssymbol:new	69, 76	\@Glsxtr@p@acrlong@:new	39
\GLSaccesssymbolplural:new	70, 76	\@Glsxtr@p@acrlongpl@:new	39
\GLSaccessstext:new	66, 75	\@Glsxtr@p@acrshort@:new	39
\glsentryfmt:moved \glssetabbrvfmt		\@Glsxtr@p@acrshortpl@:new	39
from \glsxtrabbrvfmt to here	20	\@Glsxtr@p@long@:new	38
\GlsXtrEnableInitialTagging:new ..	92	\@Glsxtr@p@longpl@:new	38
\glsxtrfieldtitlecase:new	83	\@Glsxtr@p@plural@:new	37
\GlsXtrFormatLocationList:new ...	20	\@Glsxtr@p@short@:new	37
\glsxtrnewabbrevpresetkeyhook:		\@Glsxtr@p@shortpl@:new	38
new	100	\@Glsxtr@p@text@:new	37
\glsxtrtagfont:new	93	\@Glsxtrpl:new	17
\KV@printgloss@nonumberlist:added	20	\@alt@gls@hyp@opt:new	35
\mfu@checkword@do:added	92	\@gls@alt@hyp@opt:new	35
\setabbreviationstyle:added check		\@gls@alt@hyp@opt@char:new	35
for post-definition style switch	114	\@gls@alt@hyp@opt@keys:new	35
0.5.3 (2015-12-09)		\@gls@increment@currunitcount:	
\@glsxtr@autoindex@at:new	89	new	48
\@glsxtr@autoindex@encap:new	89	\@gls@local@increment@currunitcount:	
\@glsxtr@autoindex@esc:new	90	new	48
\@glsxtr@autoindex@level:new	90	\@gls@setdefault@glslink@opts:	
\@glsxtr@autoindex@setname:new ..	88	new	33
\@glsxtr@doabbreviationsdef:new ..	6	\@glsxtr:new	16
General: removed \GlsXtrNoGlsWarningNoAutoMakeIndex		\@glsxtr@addunitcounter:new	47
.....	62	\@glsxtr@currunitcount:new	49
\glsdescwidth:added	19	\@glsxtr@ifunitcounter:new	47
\glspagelistwidth:added	19	\@glsxtr@p@acrlong@:new	39
\glsxtrdoautoindexname:new	87	\@glsxtr@p@acrlongpl@:new	39
\glsxtrpostnamehook:new	86	\@glsxtr@p@acrshort@:new	39
\if@glsxtr@format@override:new ..	87	\@glsxtr@p@acrshortpl@:new	39
\ProvidesGlossariesExtraLang:new	153	\@glsxtr@p@long@:new	38
\RequireGlossariesExtraLang:new	153	\@glsxtr@p@longpl@:new	38

\@glstr@p@plural@: new	37	\@glstr@idx@entrynumberlist: new	59
\@glstr@p@short@: new	37	\@glstr@noidx@displaynumberlist:	
\@glstr@p@shortpl@: new	38	new	58
\@glstr@p@text@: new	37	\@glstr@noidx@entrynumberlist:	
\@glstr@prevunitcount: new	49	new	59
\@glstr@unitcountlist: new	47	\@glstr@noidx@numberlistloop:	
\@glstrpl: new	16	new	59
\@newglossaryentryposthook: added		\@glstr@reg@glosslist: new	55
empty see value if not set and added		\makeglossaries: new	56
'see' to field key map	13	1.0 (2016-01-24)	
\@sGlsXtrEnableOnTheFly: new	15	\@glstr@autoindexcrossrefs: new ..	6
\cGlsformat: added	46	1.01 (2016-02-02)	
\cGlsformat: added	46	\glstrdiscardperiod: added check	
\cGlsplformat: added	47	for first use	95
\cGlsplformat: added	46	short-desc: fixed typo in \glstrinlinefullformat	
\gl:disablehyper: added	36	and added missing second argument	124
\gl:sdohyperlink: added	35	1.02 (2016-04-25)	
\gl:sdohyperlink: added	36	\@glstr@current@style: new	18
\gl:senableentryunitcount: new	49	\Glsfmtfull: new	152
\gl:shasattribute: added check for en-		\Glsfmtfull: new	152
try's existence	79	\Glsfmtfullpl: new	152
\gl:sifattribute: added check for en-		\Glsfmtfullpl: new	152
try's existence	79	\Glsfmtlong: new	151
\gl:spostlinkhook: added existence		\Glsfmtlong: new	150
check	94	\Glsfmtlongpl: new	151
\Glsxtr: new	16	\Glsfmtlongpl: new	151
\glxtr: new	16	\Glsxtrheadfull: new	146
\glxtrcat: new	15	\glxtrheadfull: new	145
\glxtrdowrglossaryhook: new	35	\Glsxtrheadfullpl: new	146
\GlsXtrEnableEntryUnitCounting:		\glxtrheadfullpl: new	146
new	52	\Glsxtrheadlong: new	144
\GlsXtrEnableOnTheFly: new	15	\glxtrheadlong: new	143
\Glsxtrpl: new	17	\Glsxtrheadlongpl: new	145
\glxtrpl: new	16	\glxtrheadlongpl: new	144
\glxtrpostlocalreset: new	40	\Glsxtrtitlefull: new	146
\glxtrpostlocalunset: new	40	\glxtrtitlefull: new	145
\glxtrpostreset: new	40	\Glsxtrtitlefullpl: new	147
\glxtrpostunset: new	40	\glxtrtitlefullpl: new	146
\glxtrprotectlinks: new	36	\Glsxtrtitlelong: new	145
\GlsXtrSetAltModifier: new	35	\glxtrtitlelong: new	144
\GlsXtrSetDefaultGlsOpts: new	34	\Glsxtrtitlelongpl: new	145
\glxtrstarflywarn: new	15	\glxtrtitlelongpl: new	144
\GlsXtrWarning: new	17	\ifglxtrininsertinside: new	116
\MakeAcronymsAbbreviations: now		postfootnote: added redef of	
disables \setacronymstyle	54	\glxtrsetupfulldefs	121
0.5.5 (??)		stylemods: new	9
\@glstr@idx@displaynumberlist:			
new	58		

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	
\.	95, 160
\@@cGLS@	42, 50, 51
\@@cGLSpl@	42, 51
\@@cGLspl@	42, 50
\@@cgls@	42, 50
\@@cglspl@	42, 44, 50
\@glo@assign@sortkey	58
\@glo@no@assign@sortkey	58
\@glslocalreset	40
\@glslocalunset	40
\@glsreset	40
\@glsunset	40
\@glsxtr@autoindex@escspch	89–91
\@glsxtr@checkspch	88, 89, 91
\@glsxtr@disabledflycommand	18
\@newglossaryentry@defcounters	41
\@newglossaryentry@defunitcounters	49
\@ACRlong	37
\@ACRlongpl	37
\@ACRshort	37
\@ACRshortpl	37
\@Acrlong	37
\@Acrlongpl	37
\@Acrshort	37
\@Acrshortpl	37
\@GLS@	36, 45, 46
\@GLSdesc@	24
\@GLSpl@	36, 45, 46
\@GLSplural@	37
\@GLSsymbol@	25
\@GLStext@	37
\@GLSxtr@full	104
\@GLSxtr@fullpl	105
\@GLSxtr@p@acrlong@	37
\@GLSxtr@p@acrlongpl@	37
\@GLSxtr@p@acrshort@	37
\@GLSxtr@p@acrshortpl@	37
\@GLSxtr@p@long@	36
\@GLSxtr@p@longpl@	36
\@GLSxtr@p@plural@	36
\@GLSxtr@p@short@	36
\@GLSxtr@p@shortpl@	36
\@GLSxtr@p@text@	36
\@GLSxtrlong	36, 108
\@GLSxtrlongpl	36, 110, 111
\@GLSxtrpl	17
\@GLSxtrshort	36, 106
\@GLSxtrshortpl	36, 109
\@GLSxtr@p@acrshortpl@	37
\@GLSxtr@p@long@	36
\@GLSxtr@p@longpl@	36
\@GLSxtr@p@plural@	36
\@GLSxtr@p@short@	36
\@GLSxtr@p@shortpl@	36
\@GLSxtr@p@text@	36
\@GLSxtrlong	36, 108
\@GLSxtrlongpl	36, 110, 111
\@GLSxtrpl	17
\@GLSxtrshort	36, 106
\@GLSxtrshortpl	36, 109

\@acrlong	37	\@gls@encapchar	89
\@acrlongpl	37	\@gls@entry@count	42, 43
\@acrshort	37	\@gls@entry@field	31, 42
\@acrshortpl	37	\@gls@entry@unitcount	51
\@alt@gls@hyp@opt	35	\@gls@field@link	21–26, 31, 32, 64–72
\@auxout	43, 51, 56, 60	\@gls@hyp@opt	31, 32, 35, 45, 46, 102–111
\@cGLS	45	\@gls@hyp@opt@cs	35
\@cGLS@	42, 45, 50, 51	\@gls@increment@currcount	42
\@cGLSpl	46	\@gls@increment@currunitcount	50
\@cGLSpl@	42, 46, 51	\@gls@keymap	13, 31
\@cGLspl@	42, 50	\@gls@label	34, 114
\@cgls@	42, 45, 50	\@gls@levelchar	88
\@cglspl@	42, 50	\@gls@link	21, 27–30, 103–111
\@disable@onlypremakeg	56	\@gls@link@checkfirsthyper	55
\@do@auxoutstuff	60	\@gls@link@nocheckfirsthyper	21, 26–30, 103–111
\@do@newglossaryentry	53, 54, 100	\@gls@local@increment@currcount	42
\@empty	21, 26–30, 88, 89, 103–111	\@gls@local@increment@currunitcount	50
\@end@glsxtr@addunused	13, 14	\@gls@loclist	58, 59
\@endfortrue	114	\@gls@longpl	98–100
\@firstofone	87, 92	\@gls@noidx@nosanitizesort	57
\@firstofthree	21, 26–28, 30, 35, 103, 104, 106, 107, 109, 110	\@gls@noidx@sanitizesort	57
\@firstoftwo	21–25, 27, 28, 30, 33, 35, 96, 97, 103–105, 109–111	\@gls@noidx@loclist@finalsep	58
\@for	10, 14, 41, 52, 56, 58, 82, 92	\@gls@noidx@loclist@prev	58
\@glo@assign@sortkey	57	\@gls@noidx@loclist@sep	58
\@glo@category	47	\@gls@org@glsnoidx@displayloc	59
\@glo@countunit	47	\@gls@org@glsseeformat	59
\@glo@default@sorttype	58	\@gls@preglossaryhook	92
\@glo@label	13, 31	\@gls@quotechar	88
\@glo@name	88	\@gls@reference	56
\@glo@see	13	\@gls@setdefault@glslink@opts	34
\@glo@sort	88	\@gls@short	99
\@glo@sorttype	58	\@gls@shortpl	98–100
\@glo@tmp	31	\@gls@tmpb	91
\@glo@type	13, 53, 56, 57, 60, 63, 64	\@gls@type	58, 114
\@glo@types	80, 81	\@gls@write@entrycounts	42
\@glossary@default@style	18, 160	\@gls@write@entryunitcounts	51
\@gls@	36, 44, 45	\@gls@write@entryunitcounts@do	52
\@gls@actualchar	88	\@glsabbrv@current@abbreviation	99, 111
\@gls@alt@hyp@opt	35	\@glsacronymlists	53
\@gls@alt@hyp@opt@char	35	\@glsentry	43, 51, 52
\@gls@alt@hyp@opt@keys	35	\@glsfirstplural@	23
\@gls@automake	58	\@glslink	36
\@gls@checkedmkidx	88, 89, 91	\@glsnumberformat	87
\@gls@checkmkidxchars	88	\@glsorder	56
\@gls@codepage	60	\@glspl@	36, 44, 45
\@gls@declareoption	4	\@glsplural@	37, 66
\@gls@doautomake	58	\@glsplunc@token	97
		\@glsstyle@inline	160

<code>\@glsstyle@listdotted</code>	155	<code>\@glsxtr@format@overridetrue</code>	87
<code>\@gls@target</code>	36	<code>\@glsxtr@foundinlist</code>	97
<code>\@gls@text@</code>	37	<code>\@glsxtr@full</code>	102
<code>\@glsxtr</code>	16, 17	<code>\@glsxtr@fullpl</code>	104
<code>\@glsxtr@abbreviationsdef</code>	7, 10	<code>\@glsxtr@gobbleto@enddescspch</code>	91
<code>\@glsxtr@activate@initialtagging</code>	92, 93	<code>\@glsxtr@idx@displaynumberlist</code>	56
<code>\@glsxtr@addunitcounter</code>	47	<code>\@glsxtr@idx@entrynumberlist</code>	57
<code>\@glsxtr@addunusedxrefs</code>	13, 14	<code>\@glsxtr@ifcsstart</code>	15
<code>\@glsxtr@attrval</code>	87, 88	<code>\@glsxtr@ifpunctoken</code>	97
<code>\@glsxtr@autoindex@at</code>	88, 89	<code>\@glsxtr@ifunitcounter</code>	47
<code>\@glsxtr@autoindex@doextra@esc</code>	88	<code>\@glsxtr@insert@dots</code>	98
<code>\@glsxtr@autoindex@encap</code>	88, 89	<code>\@glsxtr@insert@dots@next</code>	99
<code>\@glsxtr@autoindex@esc</code>	88, 90, 91	<code>\@glsxtr@insertdots</code>	99
<code>\@glsxtr@autoindex@escat</code>	88, 89	<code>\@glsxtr@label</code>	14, 82
<code>\@glsxtr@autoindex@escencap</code>	89	<code>\@glsxtr@loadstyles</code>	154
<code>\@glsxtr@autoindex@esclevel</code>	89, 90	<code>\@glsxtr@noidx@displaynumberlist</code>	57
<code>\@glsxtr@autoindex@escquote</code>	88, 90	<code>\@glsxtr@noidx@entrynumberlist</code>	57
<code>\@glsxtr@autoindex@level</code>	88–90	<code>\@glsxtr@noidx@numberlistloop</code>	57
<code>\@glsxtr@autoindex@setname</code>	87	<code>\@glsxtr@notfoundinlist</code>	97
<code>\@glsxtr@autoindex@crossrefs</code>	5, 13	<code>\@glsxtr@optlist</code>	17
<code>\@glsxtr@cat</code>	41, 52, 53, 92	<code>\@glsxtr@org@Glsxtr@titlefirst</code>	137, 138
<code>\@glsxtr@csname</code>	48, 50	<code>\@glsxtr@org@Glsxtr@titlefirstplural</code>	137, 138
<code>\@glsxtr@current@style</code>	18, 160	<code>\@glsxtr@org@Glsxtr@titlefull</code>	137, 138
<code>\@glsxtr@currentunitcount</code>	48, 50	<code>\@glsxtr@org@Glsxtr@titlefullpl</code>	137, 138
<code>\@glsxtr@currunitcount</code>	49, 51	<code>\@glsxtr@org@Glsxtr@titlelong</code>	137, 138
<code>\@glsxtr@declareoption</code>	4, 6, 7, 9	<code>\@glsxtr@org@Glsxtr@titlelongpl</code>	137, 138
<code>\@glsxtr@defaultnoglossarywarning</code>	9	<code>\@glsxtr@org@Glsxtr@titleplural</code>	137, 138
<code>\@glsxtr@disabledflycommand</code>	17	<code>\@glsxtr@org@Glsxtr@titleshort</code>	137, 138
<code>\@glsxtr@do@wrindex</code>	34	<code>\@glsxtr@org@Glsxtr@titleshortpl</code>	137, 138
<code>\@glsxtr@do@glsgl@disablehyperinlist</code>	33	<code>\@glsxtr@org@Glsxtr@titletext</code>	137, 138
<code>\@glsxtr@do@titlecaps@warn</code>	83–85, 93	<code>\@glsxtr@org@MakeUppercase</code>	137, 138
<code>\@glsxtr@doabbreviationsdef</code>	7	<code>\@glsxtr@org@checkfirsthyper</code>	32, 55
<code>\@glsxtr@doaccsupp</code>	9, 10	<code>\@glsxtr@org@Glsxtr@titlefirst</code>	137, 138
<code>\@glsxtr@dostylewarn</code>	114	<code>\@glsxtr@org@Glsxtr@titlefirstplural</code>	137, 138
<code>\@glsxtr@enabletagging</code>	92	<code>\@glsxtr@org@Glsxtr@titlefull</code>	137, 138
<code>\@glsxtr@end@</code>	15	<code>\@glsxtr@org@Glsxtr@titlefullpl</code>	137, 138
<code>\@glsxtr@enddescspch</code>	88–91	<code>\@glsxtr@org@Glsxtr@titlelong</code>	137, 138
<code>\@glsxtr@entrycount@org@localreset</code>	42	<code>\@glsxtr@org@Glsxtr@titlelongpl</code>	137, 138
<code>\@glsxtr@entrycount@org@localunset</code>	42	<code>\@glsxtr@org@Glsxtr@titleplural</code>	137, 138
<code>\@glsxtr@entrycount@org@reset</code>	42	<code>\@glsxtr@org@Glsxtr@titleshort</code>	137, 138
<code>\@glsxtr@entrycount@org@unset</code>	42	<code>\@glsxtr@org@Glsxtr@titleshortpl</code>	137, 138
<code>\@glsxtr@entryunitcount@org@localreset</code>	50	<code>\@glsxtr@org@Glsxtr@titletext</code>	137, 138
<code>\@glsxtr@entryunitcount@org@localunset</code>	50	<code>\@glsxtr@org@makeglossaries</code>	55, 56
<code>\@glsxtr@entryunitcount@org@reset</code>	50	<code>\@glsxtr@org@markboth</code>	137
<code>\@glsxtr@entryunitcount@org@unset</code>	50	<code>\@glsxtr@org@markright</code>	136, 137
<code>\@glsxtr@field@linkdefs</code>	21	<code>\@glsxtr@org@newacronymstyle</code>	54, 55
<code>\@glsxtr@format@overridefalse</code>	87	<code>\@glsxtr@org@postdescription</code>	93, 94

<code>\afp</code>	7	<code>\cs</code>	45
<code>\AL</code>	7	<code>\csdef</code>	31, 32, 42, 47, 48, 50, 78, 114, 115, 121, 155
<code>\Al</code>	7	<code>\csedef</code>	48
<code>\al</code>	7	<code>\csgdef</code>	42, 48, 50, 51
<code>\ALP</code>	7	<code>\csname</code>	18, 20, 27–32, 48, 60, 63, 64, 83, 98, 103–111, 115
<code>\Alp</code>	7	<code>\csuse</code>	31, 32, 47–51, 78, 94, 95, 114
<code>\alp</code>	7	<code>\csxdef</code>	13, 48, 51
<code>\AnyTrackedLanguages</code>	153	<code>\CurrentOption</code>	10, 154
<code>\appto</code>	10, 13, 31, 34, 41, 49, 87, 96, 99	<code>\CurrentTrackedTag</code>	153
<code>\AS</code>	7	<code>\CustomAbbreviationFields</code>	100, 116–119, 121, 123–125, 127
<code>\As</code>	7		
<code>\as</code>	7		
<code>\ASP</code>	7		
<code>\Asp</code>	7		
<code>\asp</code>	7		
<code>\AtBeginDocument</code>	11, 19		
<code>\AtEndDocument</code>	13, 42, 51, 60		
B			
babel package	87, 89, 96		
<code>\begin</code>	45, 62, 155–159		
C			
category attributes:			
discardperiod	95		
entrycount	40, 41, 43, 52		
firsttuc	86		
glossdesc	83		
glossname	84		
headuc	138		
indexname	88		
indexonlyfirst	34		
insertdots	99		
nohyper	33		
nohyperfirst	119		
regular	20, 46, 116–121, 124–126		
<code>\cGLS</code>	7, 41, 52		
<code>\cGls</code>	7, 41, 52		
<code>\cglS</code>	7, 41, 52		
<code>\cGLSformat</code>	45		
<code>\cGlsformat</code>	44		
<code>\cglSformat</code>	44, 46		
<code>\cGLSpl</code>	7, 41, 52		
<code>\cGlspl</code>	7, 41, 52		
<code>\cglSpl</code>	7, 41, 52		
<code>\cGLSplformat</code>	45		
<code>\cGlsplformat</code>	44		
<code>\cglSplformat</code>	44, 46		
<code>\columnwidth</code>	19		
<code>\count@</code>	43, 51, 52		
<code>\cs</code>	45		
<code>\csdef</code>	31, 32, 42, 47, 48, 50, 78, 114, 115, 121, 155		
<code>\csedef</code>	48		
<code>\csgdef</code>	42, 48, 50, 51		
<code>\csname</code>	18, 20, 27–32, 48, 60, 63, 64, 83, 98, 103–111, 115		
<code>\csuse</code>	31, 32, 47–51, 78, 94, 95, 114		
<code>\csxdef</code>	13, 48, 51		
<code>\CurrentOption</code>	10, 154		
<code>\CurrentTrackedTag</code>	153		
<code>\CustomAbbreviationFields</code>	100, 116–119, 121, 123–125, 127		
D			
<code>\DeclareAcronymList</code>	53		
<code>\DeclareOption</code>	4, 154		
<code>\DeclareOptionX</code>	4, 10		
<code>\def</code>	11, 14–17, 19–30, 35–39, 44–46, 53, 56, 58, 64–72, 87–93, 96–100, 103–111, 114		
<code>\define@boolkey</code>	5, 33		
<code>\define@choicekey</code>	5, 8, 9		
<code>\define@key</code>	9, 31, 98		
<code>\DefineAcronymSynonyms</code>	8		
<code>\detokenize</code>	15		
<code>\dimen@</code>	55		
<code>\dimexpr</code>	19		
<code>\disable@keys</code>	6, 11, 14		
<code>\do</code>	10, 14, 41, 52, 56, 58, 82, 92		
<code>\do@glS@link@checkfirsthyper</code>	21, 26–30, 103–111		
<code>\do@glSdisablehyperinlist</code>	33		
doc package	90		
<code>\DTLifinlist</code>	56, 57		
E			
<code>\eappto</code>	10, 88, 154		
<code>\edef</code>	33, 47, 48, 50, 56, 57, 60, 88, 89, 91, 98		
<code>\else</code>	5, 6, 9, 14, 15, 20, 34, 43, 55, 57, 61–63, 87–89, 91, 97, 99, 117–127, 156–160		
<code>\emph</code>	134–136		
<code>\encapchar</code>	90		
<code>\end</code>	62, 155–159		
<code>\endcsname</code>	18, 20, 27–32, 48, 60, 63, 64, 83, 98, 103–111, 115		
entry categories:			
abbreviation	111		
general	78, 80		
index	81		
<code>\epreto</code>	88		
<code>\equal</code>	63		

etoolbox package	4	\glossarytoctitle	63
\expandafter	10, 13, 15–17, 31, 32, 35, 46, 47, 56, 57, 83, 85, 86, 88, 90, 91, 97, 99, 100	\glossentry	155–159
\expandonce	53, 88, 89	\glossentrydesc	155–160
F			
\fi	5, 6, 9, 13–15, 18–20, 34, 43, 51, 52, 55, 57, 58, 60–64, 87–89, 91, 97, 99, 116–127, 156–160	\glossentryname	155–159
\firstacronymfont	54, 55	\glossentrysymbol	156–160
\footnote	119–121	\GLS	41, 52
\forall glossaries	13, 80–82	\Gls	16, 41, 52
\forall glsentries	43, 51	\gls	16, 18, 41, 52, 61
\ForEachTrackedDialect	153	\gls@assign@field	31
\for glsentries	13, 80–82	\gls@checkseeallowed	14, 56
\forlistcsloop	52	\gls@codepage	60
\forlistloop	58, 59, 93	\gls@defdocnewglossaryentry	41, 49
\futurelet	97	\gls@defglossaryentry	16, 17
G			
\gdef	89, 90	\gls@save@numberlist	19, 20
\Genacrfullformat	54	\glsabbrvdefaultfont	102, 116, 118, 120, 122–124, 126
\genacrfullformat	54	\glsabbrvfont	37, 38, 54, 102, 106, 107, 109, 110, 112, 116, 118–124, 126–136
\GenericAcronymFields	53	\glsabbrvfont	116–119, 121
\Genplacrfullformat	54	\GLSaccessdesc	71
\genplacrfullformat	54	\Glsaccessdesc	71, 83, 91
\glo@name	85, 86	\glsaccessdesc	71, 83, 95
glossaries package	4, 5, 7, 8, 10–12, 18, 20, 32, 33, 36, 40, 45, 53, 55, 85, 93, 98, 136, 154	\GLSaccessdescplural	72
glossaries-accsupp package	9, 10, 64	\Glsaccessdescplural	72
glossaries-extra package	2, 64	\glsaccessdescplural	72
glossaries-extra-stylemods package	9, 94, 153	\GLSaccessfirst	68
\GlossariesExtraWarning	5, 6, 15, 17, 54, 62, 92, 93	\Glsaccessfirst	67
\GlossariesExtraWarningNoLine	6, 43, 52	\glsaccessfirst	67
\GlossariesWarning	58, 59, 114	\GLSaccessfirstplural	69
\GlossariesWarningNoLine	56, 60	\Glsaccessfirstplural	68
glossary styles:		\glsaccessfirstplural	68
inline	160	\Glsaccesslong	29, 101, 108, 117, 123, 126
listdotted	155	\glsaccesslong	29, 101, 107, 108, 116, 118, 120–123, 125, 126
listdottedstyle	155	\Glsaccesslongpl	30, 101, 111, 117, 123, 126
sublistdotted	155	\glsaccesslongpl	30, 101, 110, 111, 117–123, 125–127
glossary-long package	156	\GLSaccessname	65
glossary-longbooktabs package	156	\Glsaccessname	65
glossary-mcols package	154	\glsaccessname	64
glossary-tree package	154	\GLSaccessplural	67
\glossaryentrynumbers	20	\Glsaccessplural	67
\glossaryheader	155–159	\glsaccessplural	66
\glossarysection	63	\Glsaccessshort	27, 106, 112, 118, 120–122, 125
\glossarytitle	63	\glsaccessshort	26, 27, 101, 106, 107, 112, 117, 118, 120, 122–126
		\Glsaccessshortpl	28, 109, 112, 119–122, 125

<code>\glsaccesssshortpl</code>	28,	<code>\Glsentryfullpl</code>	54
101, 109, 110, 112, 117, 118, 120, 122–126		<code>\glsentryfullpl</code>	54
<code>\GLSaccessssymbol</code>	69	<code>\glsentryitem</code>	155–159
<code>\Glsaccessssymbol</code>	69, 91	<code>\Glsentrylong</code>	38, 39, 46, 74, 77
<code>\glsaccessssymbol</code>	69, 91, 95	<code>\glsentrylong</code>	
<code>\GLSaccessssymbolplural</code>	70	38, 39, 46, 73, 74, 77, 121, 150, 151
<code>\Glsaccessssymbolplural</code>	70	<code>\Glsentrylongpl</code>	38, 39, 47, 74, 77
<code>\glsaccessssymbolplural</code>	70	<code>\glsentrylongpl</code>	38, 39, 46, 74, 77, 151
<code>\GLSaccesstext</code>	66	<code>\Glsentryname</code>	23, 65, 74, 84–86
<code>\Glsaccesstext</code>	66	<code>\glsentryname</code>	23, 64, 65, 74, 88
<code>\glsaccesstext</code>	65	<code>\glsentrynumberlist</code>	57, 59
<code>\glsacrshortcutstrue</code>	8	<code>\Glsentryplural</code>	22, 66, 75
<code>\glsacspacemax</code>	55	<code>\glsentryplural</code>	22, 66, 67, 75, 149
<code>\glsadd</code>	14, 61	<code>\glsentryprevcount</code>	42, 43, 49
<code>\glsaddstoragekey</code>	78	<code>\glsentryprevmaxcount</code>	50
<code>\glsbackslash</code>	15	<code>\glsentryprevtotalcount</code>	49
<code>\glscapscase</code>	21–30, 32, 103–113	<code>\Glsentryshort</code>	37, 39, 73, 77
<code>\glscategory</code>	20, 33, 37, 38,	<code>\glsentryshort</code>	
79, 80, 83–86, 91, 94, 95, 103–107, 109, 110		37–39, 55, 72, 73, 76, 77, 147, 148
<code>\glscategorylabel</code>	33, 98–100, 121	<code>\Glsentryshortpl</code>	38, 39, 73, 77
<code>\glsclosebrace</code>	62, 63	<code>\glsentryshortpl</code>	38, 39, 73, 77, 147, 148
<code>\glscurrententrylabel</code>	93, 94	<code>\Glsentrysymbol</code>	24, 69, 76
<code>\glscustomtext</code>	21, 26–30, 103–111, 113	<code>\glsentrysymbol</code>	24, 69, 75, 76
<code>\glsdefaulttype</code>	6, 61	<code>\Glsentrysymbolplural</code>	24, 70, 76
<code>\glsdescriptionaccessdisplay</code> ..	70, 71, 83	<code>\glsentrysymbolplural</code>	24, 25, 70, 76
<code>\glsdescriptionpluralaccessdisplay</code>		<code>\Glsentrytext</code>	21, 65, 75
.....	71, 72	<code>\glsentrytext</code>	21, 65, 66, 75, 148
<code>\glsdescwidth</code>	155, 157–159	<code>\Glsentryuseri</code>	25
<code>\glsdetoklabel</code>		<code>\glsentryuseri</code>	25
... 11–13, 15, 42, 47–52, 58, 59, 83, 85, 86		<code>\Glsentryuserii</code>	25
<code>\glsdisplaynumberlist</code>	56, 58	<code>\glsentryuserii</code>	25
<code>\glsdohyperlink</code>	36	<code>\Glsentryuseriii</code>	25
<code>\glsdoifexists</code> ...	21, 26–30, 58, 59, 103–111	<code>\glsentryuseriii</code>	25
<code>\glsdoifexistsorwarn</code>	83–86, 91	<code>\Glsentryuseriv</code>	26
<code>\glsdonohyperlink</code>	36	<code>\glsentryuseriv</code>	26
<code>\glsdosanitizesort</code>	57	<code>\Glsentryuserv</code>	26
<code>\glsenableentrycount</code>	41, 43, 51	<code>\glsentryuserv</code>	26
<code>\glsenableentryunitcount</code>	42, 52	<code>\Glsentryuservi</code>	26
<code>\glsentrycurrcount</code>	42, 43, 49	<code>\glsentryuservi</code>	26
<code>\Glsentrydesc</code>	23, 71, 76, 84	<code>\glsfieldxdef</code>	82
<code>\glsentrydesc</code>	23, 70, 71, 76, 84	<code>\GLSfirst</code>	142
<code>\Glsentrydescplural</code>	24, 72, 76	<code>\Glsfirst</code>	142, 143
<code>\glsentrydescplural</code>	24, 71, 72, 76	<code>\glsfirst</code>	142
<code>\Glsentryfirst</code>	22, 46, 67, 75	<code>\glsfirstabbrvdefaultfont</code>	
<code>\glsentryfirst</code> 21, 22, 46, 67, 68, 75, 149, 150		102, 116, 118, 120, 122–124, 126
<code>\Glsentryfirstplural</code>	23, 47, 68, 75	<code>\glsfirstabbrvfont</code>	54, 101, 116–126
<code>\glsentryfirstplural</code> 22, 23, 46, 68, 75, 150		<code>\glsfirstaccessdisplay</code>	67, 68
<code>\Glsentryfull</code>	54	<code>\glsfirstlongdefaultfont</code>	
<code>\glsentryfull</code>	54	102, 116, 118, 120, 122–124, 126

<code>\glsfirstlongfont</code>	101, 116–127	<code>\glsnumlistlastsep</code>	58
<code>\GLSfirstplural</code>	143	<code>\glsnumlistsep</code>	58
<code>\Glsfirstplural</code>	143	<code>\glsopenbrace</code>	62, 63
<code>\glsfirstplural</code>	143	<code>\glsorder</code>	56
<code>\glsfirstpluralaccessdisplay</code>	68	<code>\glspagelistwidth</code>	155, 157–159
<code>\glsforeachincategory</code>	114	<code>\GLSpl</code>	41, 52
<code>\glsgenentryfmt</code>	20	<code>\Glspl</code>	17, 41, 52
<code>\glsgetattribute</code>	43, 47–49, 87	<code>\glspl</code>	17, 41, 52
<code>\glsgetcategoryattribute</code>	79	<code>\GLSplural</code>	141, 142
<code>\glsgroupheading</code>	155–159	<code>\Glsplural</code>	142
<code>\glsgroupskip</code>	156–160	<code>\glsplural</code>	141
<code>\glshasattribute</code> .	43, 48, 50, 52, 87, 116–121	<code>\glspluralaccessdisplay</code>	66, 67
<code>\glshascategoryattribute</code>	79	<code>\glspluralsuffix</code>	99,
<code>\glshypernumber</code>	87		102, 116, 118, 120, 122–124, 126, 127, 131
<code>\glsifattribute</code>		<code>\glspostdescription</code>	93, 155–160
....	33, 34, 81, 83–85, 93, 95, 96, 139–146	<code>\glspostinline</code>	160
<code>\glsifcategory</code>	80	<code>\glspostlinkhook</code>	21, 27–30, 103–111
<code>\glsifcategoryattribute</code>	33, 79, 80, 99, 100	<code>\glsprestandardsort</code>	57
<code>\glsifplural</code>	21–30, 95, 96, 103–112	<code>\glseeformat</code>	59
<code>\glsifregular</code>	20, 46, 47	<code>\glsetabbrfmt</code>	
<code>\glsifregularcategory</code>	80	..	20, 37, 38, 83–86, 91, 103–107, 109, 110
<code>\glsinlinedescformat</code>	160	<code>\glsetattribute</code>	116–121, 123, 124, 126, 127
<code>\glsinlinesubdescformat</code>	160	<code>\glsetcategoryattribute</code>	
<code>\glsinsert</code>	21, 26–30, 103–113	41, 53, 55, 79–82, 92
<code>\glskeylisttok</code>	53, 99, 100	<code>\glsshortaccessdisplay</code>	72, 73
<code>\glslabel</code>	20, 32, 33, 94, 95, 112, 113, 121	<code>\glsshortpltok</code> ..	100, 116–119, 121, 123, 124
<code>\glslabeltok</code>		<code>\glsshortpluralaccessdisplay</code>	73
..	53, 99, 100, 116–121, 123, 124, 126, 127	<code>\glsshorttok</code>	
<code>\glsletentryfield</code>	88	..	53, 99, 100, 116–119, 121, 123, 124, 127
<code>\glslink</code>	54	<code>\glssubentryitem</code>	155–160
<code>\glslink options</code>		<code>\glssymbolaccessdisplay</code>	69
format	87	<code>\glssymbolpluralaccessdisplay</code>	70
noindex	33	<code>\glstarget</code>	155–160
<code>\glslinkcheckfirsthyperhook</code>	33	<code>\GLStext</code>	140, 141
<code>\glslinkvar</code>	35	<code>\Glstext</code>	141
<code>\glslistdottedwidth</code>	155	<code>\glstext</code>	140, 141
<code>\glslongaccessdisplay</code>	73, 74	<code>\glstextaccessdisplay</code>	65, 66
<code>\glslongpltok</code> ...	100, 116–119, 121, 125–127	<code>\glstextup</code>	127
<code>\glslongpluralaccessdisplay</code>	74	<code>\glstype</code>	27–30, 103–111
<code>\glslongtok</code>	53, 99, 100, 116–119, 121, 123–127	<code>\glunset</code>	14, 44, 45
<code>\glsnameaccessdisplay</code>	64, 65, 84–86	<code>\glswrite</code>	56
<code>\glsnamefont</code>	84–86	<code>\Glsxtr</code>	17
<code>\glsnoidxdisplayloc</code>	59	<code>\glxtr</code>	17
<code>\glsnoidxdisplayloclisthandler</code>	58	<code>\glxtr@addunused</code>	13
<code>\glsnoidxloclist</code>	59	<code>\glxtr@applyabbrfmt</code>	111
<code>\glsnoidxnumberlistloophandler</code>	59	<code>\glxtr@applyabbrvstyle</code>	98, 99, 114
<code>\glsnonumberlistfalse</code>	20	<code>\glxtr@doooption</code>	4, 6, 10
<code>\glsnonumberlisttrue</code>	20	<code>\glxtr@ifnextpunc</code>	97
<code>\glsnumberlistloop</code>	57	<code>\glxtr@ifpunctoken</code>	97

<code>\glxtr@keylist</code>	16, 17	<code>\Glsxtrheadshort</code>	137
<code>\glxtr@next</code>	97	<code>\glxtrheadshort</code>	137
<code>\glxtr@orgmakenoidxglossaries</code>	14	<code>\Glsxtrheadshortpl</code>	138
<code>\glxtr@punc</code>	96, 97	<code>\glxtrheadshortpl</code>	137
<code>\glxtr@warnonexistsordo</code>	5, 12	<code>\Glsxtrheadtext</code>	138
<code>\glxtrabbrvtype</code>	6, 100	<code>\glxtrheadtext</code>	138
<code>\glxtraddallcrossrefs</code>	13	<code>\glxtrtrifcounttrigger</code>	44, 45
<code>\glxtrcat</code>	16, 17	<code>\glxtrtrifemptyglossary</code>	63
<code>\GlsXtrDefineAbbreviationShortcuts</code>	8, 9	<code>\glxtrtrifindexing</code>	34
<code>\GlsXtrDefineOtherShortcuts</code>	8, 9	<code>\glxtrtrifnextpunc</code>	96, 97
<code>\glxtrdiscardperiod</code>	94	<code>\glxtrtrifperiod</code>	95, 96
<code>\glxtrdoautoindexname</code>	34, 87	<code>\glxtrtrifwasfirstuse</code>	21– 23, 26–30, 33, 95, 103, 106–111, 121, 122
<code>\glxtrdopostpunc</code>	121	<code>\Glsxtrinlinefullformat</code>	102, 103, 115, 121–123, 125, 126, 152
<code>\glxtrdowrglossaryhook</code>	34	<code>\glxtrinlinefullformat</code>	102–104, 115, 120, 122–124, 126, 152
<code>\GlsXtrEnableEntryCounting</code>	52	<code>\Glsxtrinlinefullplformat</code>	102, 105, 115, 121–123, 125, 126, 152
<code>\GlsXtrEnableEntryUnitCounting</code>	41	<code>\glxtrinlinefullplformat</code>	101, 102, 104, 105, 115, 120, 122, 123, 125, 126, 152
<code>\GlsXtrEnableOnTheFly</code>	15, 18	<code>\glxtrinsertinsidefalse</code>	116
<code>\glxtrfieldtitlecase</code>	83–85	<code>\GLSxtrlong</code>	7, 144
<code>\GlsXtrFormatLocationList</code>	20	<code>\Glsxtrlong</code>	7, 144, 145
<code>\GLSxtrfull</code>	7, 145, 146	<code>\glxtrlong</code>	7, 144
<code>\Glsxtrfull</code>	7, 146	<code>\GLSxtrlongpl</code>	7, 144, 145
<code>\glxtrfull</code>	7, 145	<code>\Glsxtrlongpl</code>	7, 145
<code>\Glsxtrfullformat</code>	101, 113, 115, 117, 118, 120, 122, 124–126	<code>\glxtrlongpl</code>	7, 144
<code>\glxtrfullformat</code>	101, 113, 115–120, 122, 124–126	<code>\glxtrmarkhook</code>	136, 137
<code>\GLSxtrfullpl</code>	7, 146, 147	<code>\glxtrnewabbrevpresetkeyhook</code>	100
<code>\Glsxtrfullpl</code>	7, 147	<code>\glxtrnewnumber</code>	7
<code>\glxtrfullpl</code>	7, 146	<code>\glxtrnewsymbol</code>	7
<code>\Glsxtrfullplformat</code>	101, 113, 115, 117, 119, 120, 122, 124–126	<code>\glxtrNoGlossaryWarning</code>	9, 60
<code>\glxtrfullplformat</code>	113, 115, 117, 118, 120, 122, 124–126	<code>\GlsXtrNoGlsWarningAutoMake</code>	63
<code>\glxtrfullsep</code>	101, 116–123, 125, 126	<code>\GlsXtrNoGlsWarningBuildInfo</code>	64
<code>\glxtrngenabbrvfmt</code>	20	<code>\GlsXtrNoGlsWarningCheckFile</code>	63
<code>\Glsxtrheadfirst</code>	138	<code>\GlsXtrNoGlsWarningEmptyMain</code>	63
<code>\glxtrheadfirst</code>	138	<code>\GlsXtrNoGlsWarningEmptyNotMain</code>	63
<code>\Glsxtrheadfirstplural</code>	138	<code>\GlsXtrNoGlsWarningEmptyStart</code>	63
<code>\glxtrheadfirstplural</code>	138	<code>\GlsXtrNoGlsWarningHead</code>	63
<code>\Glsxtrheadfull</code>	138	<code>\GlsXtrNoGlsWarningMisMatch</code>	64
<code>\glxtrheadfull</code>	138	<code>\GlsXtrNoGlsWarningNoOut</code>	64
<code>\Glsxtrheadfullpl</code>	138	<code>\GlsXtrNoGlsWarningTail</code>	64
<code>\glxtrheadfullpl</code>	138	<code>\Glsxtrpl</code>	17
<code>\Glsxtrheadlong</code>	138	<code>\glxtrpl</code>	17
<code>\glxtrheadlong</code>	138	<code>\glxtrpostdescription</code>	81, 94, 160
<code>\Glsxtrheadlongpl</code>	138	<code>\glxtrpostlink</code>	94
<code>\glxtrheadlongpl</code>	138	<code>\glxtrpostlinkendsentence</code>	94
<code>\Glsxtrheadplural</code>	138	<code>\glxtrpostlinkhook</code>	94
<code>\glxtrheadplural</code>	138		

<code>\NoCaseChange</code>	139–146	<code>\PackageError</code> ...	5, 10, 14, 17, 18, 31, 32, 41, 42, 49, 51, 52, 54, 56, 57, 114, 115, 154
<code>\noexpand</code>	10, 53, 60, 88, 89, 100, 154	<code>\PackageWarning</code>	6
<code>\nofiles</code>	63	<code>\PackageWarningNoLine</code>	6
<code>\noindent</code>	63	<code>\par</code>	63, 64
<code>\nopostdesc</code>	16, 17, 81	<code>\PassOptionsToPackage</code>	4
<code>\nr</code>	5, 8, 9	<code>\preto</code>	33
<code>\ns@GLSxtrfull</code>	104	<code>\printabbreviations</code>	6
<code>\ns@GLsxtrfull</code>	103	<code>\printglossaries</code>	56, 62
<code>\ns@glxtrfull</code>	102	<code>\printglossary</code>	6, 56, 62
<code>\ns@GLSxtrfullpl</code>	105	<code>\printglossary options</code>	
<code>\ns@GLsxtrfullpl</code>	104	<code>nonumberlist</code>	20
<code>\ns@glxtrfullpl</code>	104	<code>\printnoidxglossaries</code>	62
<code>\ns@GLSxtrlong</code>	108	<code>\printnoidxglossary</code>	62
<code>\ns@GLsxtrlong</code>	107, 108	<code>\printnumbers</code>	7, 82
<code>\ns@glxtrlong</code>	107	<code>\printsymbols</code>	7, 81
<code>\ns@GLSxtrlongpl</code>	111	<code>\ProcessOptions</code>	154
<code>\ns@GLsxtrlongpl</code>	110	<code>\ProcessOptionsX</code>	10
<code>\ns@glxtrlongpl</code>	110	<code>\protect</code> 74–77, 101, 116–121, 123–133, 139–146	
<code>\ns@GLSxtrshort</code>	107	<code>\protected@edef</code>	18, 53, 87, 100
<code>\ns@GLsxtrshort</code>	106	<code>\protected@write</code>	56
<code>\ns@glxtrshort</code>	105, 106	<code>\providecommand</code> ...	6, 32, 43, 51, 56, 60, 154
<code>\ns@GLSxtrshortpl</code>	109	<code>\ProvidesFile</code>	153
<code>\ns@GLsxtrshortpl</code>	109	<code>\ProvidesPackage</code>	4, 154
<code>\ns@glxtrshortpl</code>	108		
<code>\null</code>	9		
<code>\number</code>	48–51		
<code>\numexpr</code>	48, 51		

O

<code>\or</code>	5, 8
<code>\org@glossaryentrynumbers</code>	19, 20

P

<code>\p@gl@s@hyp@opt</code>	35
package options:	
<code>abbreviations</code>	6, 7
<code>accsupp</code>	9, 64
<code>acronym</code>	6
<code>automake</code>	58, 61
<code>docdef</code>	14, 41, 49
<code>nonumberlist</code>	19
<code>numbers</code>	7
<code>shortcuts</code>	8
<code>all</code>	8
<code>false</code>	8
<code>none</code>	8
<code>true</code>	8
<code>symbols</code>	7, 81
<code>undefaction</code>	11, 12
<code>warn</code>	5

Q

<code>\quotechar</code>	90
-------------------------------	----

R

<code>\raggedright</code>	157, 159
<code>\relax</code>	5, 7–10, 14, 18, 19, 35, 42, 43, 51, 56, 59, 88, 89, 91, 93, 95, 98, 99, 160
<code>relese package</code>	130
<code>\renewcommand</code> ..	5, 6, 8–12, 14, 15, 17, 18, 20, 21, 31–36, 40–43, 46, 47, 49–58, 60, 81, 83–86, 91–94, 102, 115–137, 155–160
<code>\renewenvironment</code>	155–159
<code>\renewglossarystyle</code>	155–159
<code>\RequireGlossariesExtraLang</code>	153
<code>\RequirePackage</code>	4, 9, 10, 154
<code>\reserved@a</code>	97
<code>\reserved@b</code>	97
<code>\reserved@d</code>	97
<code>\RestoreAcronyms</code>	54, 55

S

<code>\s@gl@s@hyp@opt</code>	35
<code>\s@glxtr@enabletagging</code>	92
<code>\setabbreviationstyle</code>	54, 117, 124
<code>\setacronymstyle</code>	54, 55

<code>\SetGenericNewAcronym</code>	55	<code>\theindex</code>	87
<code>\setglossarystyle</code>	155, 160	<code>\this@dialect</code>	153
<code>\setkeys</code>	10, 34, 53, 99, 100	<code>\toks@</code>	91
<code>\setlength</code>	19		
<code>\settowidth</code>	55		
<code>\setupglossaries</code>	4, 10		
<code>\sfcode</code>	95, 160		
<code>\space</code>	5, 15, 18, 41–43, 49, 51, 52, 54–57, 60, 63, 95, 101, 160		
<code>\spacefactor</code>	95, 99, 160		
<code>\string</code>	5, 14, 15, 18, 31, 32, 41–43, 49, 51, 52, 54–57, 60–63, 88		
<code>\strut</code>	155–160		
<code>\subglossentry</code>	155–160		
	T		
<code>\tablehead</code>	158, 159		
<code>\tabletail</code>	158, 159		
<code>\tabularnewline</code>	155–160		
<code>\TeX</code>	61		
<code>\texorpdfstring</code>	147–152		
textcase package	136		
<code>\textsc</code>	127		
<code>\textsmaller</code>	130		
<code>\texttt</code>	61–63		
<code>\the</code>	53, 91, 100, 116–121, 123–127		
		U	
		<code>\undef</code>	92
		<code>\underline</code>	93
		<code>\unskip</code>	14, 155
		<code>\usepackage</code>	62, 63
		V	
		<code>\val</code>	5, 8, 9
		W	
		<code>\warn@nomakeglossaries</code>	56
		<code>\warn@noprintglossary</code>	56
		<code>\write</code>	43, 51, 56, 60
		X	
		<code>\xcapitalisewords</code>	83
		<code>\xifinlist</code>	47
		xindy	55
		xkeyval package	4
		<code>\XKV@checkchoice</code>	20
		<code>\XKV@plfalse</code>	20
		<code>\XKV@resa</code>	20
		<code>\XKV@sttrue</code>	20