

`glossaries-extra.sty v1.17: documented code`

Nicola L.C. Talbot

Dickimaw Books

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

2017-08-09

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)	5
1.1	Package Initialisation and Options	5
1.2	Extra Utilities	20
1.3	Modifications to Commands Provided by glossaries	26
1.3.1	Existence Checks	30
1.3.2	Document Definitions	37
1.3.3	Existing Glossary Style Modifications	42
1.3.4	Entry Formatting, Hyperlinks and Indexing	46
1.3.5	Entry Counting	79
1.3.6	Acronym Modifications	92
1.3.7	Indexing and Displaying Glossaries	95
1.4	Integration with glossaries-accsupp	117
1.5	Categories	128
1.6	Abbreviations	151
1.6.1	Abbreviation Styles Setup	170
1.6.2	Predefined Styles (Default Font)	173
1.6.3	Predefined Styles (Small Capitals)	188
1.6.4	Predefined Styles (Fake Small Capitals)	202
1.6.5	Predefined Styles (Emphasized)	215
1.6.6	Predefined Styles (User Parentheses Hook)	236
1.6.7	Predefined Styles (Hyphen)	245
1.6.8	Predefined Styles (No Short on First Use)	259
1.7	Using Entries in Headings	261
1.8	Multi-Lingual Support	279
2	Style Adjustments (glossaries-extra-stylemods.sty)	280
2.1	Package Initialisation	280
2.2	List-Like Styles	281
2.3	Longtable Styles	281
2.4	Long Ragged Styles	282
2.5	Supertabular Styles	284
2.6	Super Ragged Styles	285
2.7	Inline Style	286
2.8	Tree Styles	286
	Glossary	298
	Change History	299

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2017/08/09 v1.17 (NLCT)]
```

Requires xkeyval to define package options.

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

Requires etoolbox package.

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

Has glossaries already been loaded?

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

```
6 {%
```

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

```
7 \newcommand{\glxstr@doption}[1]{\setupglossaries{#1}}%
```

```
8 \let\@glxstr@declareoption\@gls@declareoption
```

```
9 }
```

```
10 {%
```

Not already loaded, so pass options to glossaries.

```
11 \newcommand{\glxstr@doption}[1]{%
```

```
12 \PassOptionsToPackage{#1}{glossaries}%
```

```
13 }%
```

Set the defaults.

```
14 \PassOptionsToPackage{toc}{glossaries}
```

```
15 \PassOptionsToPackage{nopostdot}{glossaries}
```

```
16 \PassOptionsToPackage{noredefwarn}{glossaries}
```

```
17 \@ifpackageloaded{polyglossia}%
```

```
18 {}%
```

```
19 {%
```

```
20 \@ifpackageloaded{babel}%
```

```
21 {\PassOptionsToPackage{translate=babel}{glossaries}}%
```

```
22 {}%
```

```
23 }%
```

```
24 \newcommand*{\@glxstr@declareoption}[2]{%
```

```
25 \DeclareOptionX{#1}{#2}%
```

```
26 \DeclareOption{#1}{#2}%
```

```
27 }
```

```
28 }
```

Declare package options.

`\glstrundefaction` Determines what to do if an entry hasn't been defined. The two arguments are the error or warning message and the help message if an error should be produced.

```

29 \newcommand*{\glstrundefaction}[2]{%
30   \@glstrundeftag\PackageError{glossaries-extra}{#1}{#2}%
31 }

```

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

```

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

```

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

```

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

```

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

`\warn@undefaction` This is how `\glstrundefaction` should behave if `undefaction=warn` is set.

```

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

```

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

```

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

```

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

```

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

```

`\f@for@gl@sentries`

```

47 \newcommand*{\@glstr@redef@for@gl@sentries}{}

```

`\f@for@gl@sentries`

```

48 \newcommand*{\@glstr@do@redef@for@gl@sentries}{%
49   \renewcommand*{\for@gl@sentries}[3][\gl@defaulttype]{%
50     \edef\@glo@list{\csname glolist@##1\endcsname}%
51     \ifdefstring{\@glo@list}{,}%
52     {%
53       \GlossariesExtraWarning{No entries defined in glossary '#1'}%
54     }%
55     {%
56       \@for##2:=\@glo@list\do

```

```

57      {%
58      \ifdefempty{##2}{-}{##3}%
59      }%
60      }%
61      }%
62 }%

63 \define@choicekey{glossaries-extra.sty}{undefaction}[\val\nr]%
64 {warn,error}%
65 {%
66   \ifcase\nr\relax
67     \let\glstrundefaction\@glstrwarn@undefaction
68     \let\glstrwarnonexistssordo\@glstrwarn@onexistssordo
69     \let\@glstr@redef@for@glstentries\@glstr@do@redef@for@glstentries
70   \or
71     \let\glstrundefaction\@glstrerr@undefaction
72     \let\glstrwarnonexistssordo\@gobble
73     \let\@glstr@redef@for@glstentries\relax
74   \fi
75 }

```

To assist bib2gls, v1.08 introduces the record option, which will write information to the aux file whenever an entry needs to be indexed.

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

```
76 \newcommand*{\@glstr@record}[3]{}
```

`\glstr@recordsee` Does nothing by default.

```
77 \newcommand*{\glstr@recordsee}[2]{}
```

`@@glstr@record` This is the actual code that does the recording. The first argument is the option list (as passed in the first optional argument to commands like `\gls`). This allows the `noindex` setting to be picked up. The third argument is the key family (`glslink` in most cases, `glossadd` for `\glsadd`).

```

78 \newcommand*{\@@glstr@record}[3]{%
79   \begingroup
80   \def\@glsnumberformat{glsnumberformat}%
81   \def\@glstr@thevalue{}%
82   \def\@glstr@theHvalue{\@glstr@thevalue}%
83   \ifcsdef{glo@#2@counter}%
84   {%
85     \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
86   }%
87   {%

```

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

```

88     \def\@gls@counter{page}%
89   }%
90   \setkeys{#3}{#1}%

```

```

91 \ifKV@glslink@noindex
92 \else
93 \glswriteentry{#2}%
94 {%

```

Check if the value has been set.

```

95 \ifdefempty{\@glstr@thevalue}%
96 {%

```

Save the entry counter.

```

97 \glstr@saveentrycounter

```

Temporarily redefine \@do@wrglossary so we can use \glstr@@do@wrglossary.

```

98 \let\@do@wrglossary\glstr@dorecord
99 }%
100 {%

```

the value has been set, so there's no need to defer writing the location value.

```

101 \let\theHglentrycounter\glstr@thevalue
102 \def\theHglentrycounter{\@glstr@theHvalue}%
103 \let\@do@wrglossary\glstr@dorecordnodefer
104 }%
105 \glstr@@do@wrglossary{#2}%
106 }%
107 \fi
108 \endgroup
109 }

```

glstr@dorecord

```

110 \newcommand*\@glstr@dorecord{%
111 \protected@write\@auxout{\let\@glslocref\relax}{\string\glstr@record
112 {\@gls@label}{\@gls@counterprefix}{\@gls@counter}{\@glsnumberformat}%
113 {\@glslocref}}}%
114 \@glstr@counterrecordhook
115 }

```

dorecordnodefer As above, but don't defer expansion of location.

```

116 \newcommand*\@glstr@dorecordnodefer{%
117 \protected@write\@auxout{}{\string\glstr@record
118 {\@gls@label}{\@gls@counterprefix}{\@gls@counter}{\@glsnumberformat}%
119 {\@glslocref}}}%
120 \@glstr@counterrecordhook
121 }

```

r@recordcounter

```

122 \newcommand*\@glstr@recordcounter{%
123 \@glstr@noop@recordcounter
124 }

```

p@recordcounter


```

125 \newcommand*{\@glxtr@noop@recordcounter}[1]{%
126   \PackageError{glossaries-extra}{\string\GlsXtrRecordCounter\space
127     requires record=only or record=alsoindex package option}{}%
128 }

p@recordcounter

129 \newcommand*{\@glxtr@op@recordcounter}[1]{%
130   \eappto\@glxtr@counterrecordhook{\noexpand\@glxtr@docounterrecord{#1}}%
131 }

lsxtr@recordsee Deal with \glssee in record mode.
132 \newcommand*{\@glxtr@recordsee}[2]{%
133   \def\@gls@xref{#2}%
134   \@onelevel@sanitize\@gls@xref
135   \protected@write\@auxout{}\string\glxtr@recordsee{#1}{\@gls@xref}}%
136 }

srtglossaryunit

137 \newcommand{\printunsrtglossaryunit}{%
138   \print@noop@unsrtglossaryunit
139 }

tr@setup@record Initialise.
140 \newcommand*{\glxtr@setup@record}{}

aveentrycounter Only store the entry counter information if the indexing is on.
141 \newcommand*{\glxtr@indexonly@saveentrycounter}{%
142   \ifKV@glslink@noindex
143   \else
144     \glxtr@saveentrycounter
145   \fi
146 }

addloclistfield

147 \newcommand*{\glxtr@addloclistfield}{%
148   \key@ifundefined{glossentry}{loclist}%
149   {%
150     \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
151     \appto\@gls@keymap{,{loclist}{loclist}}%
152     \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
153     \appto\@newglossaryentryposthook{%
154       \gls@assign@field{\@glo@label}{loclist}{\@glo@loclist}%
155     }%
156     \glssetnoexpandfield{loclist}%
157   }%
158   {%

```

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

```

159 \key@ifundefined{glossentry}{location}%
160 {%
161   \define@key{glossentry}{location}{\def\@glo@location{##1}}%
162   \appto\@gls@keymap{,{location}{location}}%
163   \appto\@newglossaryentryprehook{\def\@glo@location{}}%
164   \appto\@newglossaryentryposthook{%
165     \gls@assign@field{\@glo@label}{location}{\@glo@location}%
166   }%
167   \glssetnoexpandfield{location}%
168 }%
169 {}%
```

Add a key to store the group heading.

```

170 \key@ifundefined{glossentry}{group}%
171 {%
172   \define@key{glossentry}{group}{\def\@glo@group{##1}}%
173   \appto\@gls@keymap{,{group}{group}}%
174   \appto\@newglossaryentryprehook{\def\@glo@group{}}%
175   \appto\@newglossaryentryposthook{%
176     \gls@assign@field{\@glo@label}{group}{\@glo@group}%
177   }%
178   \glssetnoexpandfield{group}%
179 }%
180 {}%
181 }
```

Now define the record package option.

```

182 \define@choicekey{glossaries-extra.sty}{record}[\val\nr]%
183 {off,only,alsoindex}%
184 [only]%
185 {%
186   \ifcase\nr\relax
```

Don't record.

```

187   \def\glsxtr@setup@record{%
188     \renewcommand*{\@do@seeglossary}{\@glsxtr@org@doseeglossary}%
189     \renewcommand*{\@glsxtr@record}[3]{}%
190     \let\@do@wrglossary\glsxtr@do@wrglossary
191     \let\@gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
192     \let\glsxtrundefaction\@glsxtr@err@undefaction
193     \let\glsxtr@warnonexistsordo\@gobble
194     \let\@glsxtr@recordcounter\@glsxtr@noop@recordcounter
195     \def\printunsrtglossaryunit{\print@noop@unsrtglossaryunit}%
196     \undef\glsxtrsetaliasnoindex
197   }%
198   \or
```

Only record (don't index).

```

199   \def\glsxtr@setup@record{%
```

```

200 \glsxtr@autoseeindexfalse
201 \let\@do@seeglossary\glsxtr@recordsee
202 \let\glsxtr@record\@glsxtr@record
203 \let\@do@wrglossary\gobble
204 \let\gls@saveentrycounter\relax
205 \let\glsxtrundefaction\glsxtr@warn@undefaction
206 \let\glsxtr@warnonexistssordo\glsxtr@warn@onexistssordo
207 \glsxtr@addloclistfield
208 \renewcommand*{\@glsxtr@autoindexcrossrefs}{}%
209 \let\@glsxtr@recordcounter\glsxtr@op@recordcounter
210 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%

```

Switch off the index suppression for aliased entries. (bib2gls will deal with them.)

```

211 \def\glsxtrsetaliasnoindex{%

```

\gls@setupsort@none was only introduced to glossaries v4.30, so it may not be available. If it's defined, use it to remove the unnecessary overhead of escaping and sanitizing the sort value.

```

212 \ifdef\gls@setupsort@none{\@gls@setupsort@none}{}%
213 }%
214 \or

```

Record and index.

```

215 \def\glsxtr@setup@record{%
216 \renewcommand*{\@do@seeglossary}{\@glsxtr@org@doseeglossary}%
217 \let\glsxtr@record\@glsxtr@record
218 \let\@do@wrglossary\glsxtr@do@wrglossary
219 \let\gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
220 \let\glsxtrundefaction\glsxtr@warn@undefaction
221 \let\glsxtr@warnonexistssordo\glsxtr@warn@onexistssordo
222 \glsxtr@addloclistfield
223 \let\@glsxtr@recordcounter\glsxtr@op@recordcounter
224 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
225 \undef\glsxtrsetaliasnoindex
226 }%
227 \fi
228 }

```

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

```

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

```

Need to provide conditional commands that are backward compatible:

```

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

```

lsxtrdocdeftrue

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

lsxtrdocdeffalse

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

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

```
233 \define@choicekey{glossaries-extra.sty}{docdef}{\val\nr}%
234 {false,true,restricted}[true]%
235 {%
236   \@glxtr@docdefval=\nr\relax
237   \ifnum\@glxtr@docdefval=2\relax
238     \renewcommand*{\@glsdofexistsorwarn}{\glsdofexists}%
239   \fi
240 }
```

ocdefrestricted

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

oifexistsorwarn

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

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

indexcrossrefs

Automatically index cross references at the end of the document

```
243 \define@boolkey{glossaries-extra.sty}{@glxtr}{indexcrossrefs}[true]{%
244   \if@glxtrindexcrossrefs
245   \else
246     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
247   \fi
248 }
```

Switch off since this can increase the build time.

```
249 \@glxtrindexcrossrefsfalse
```

But allow see key to switch it on automatically.

oindexcrossrefs

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

autoseeindex

Provide a boolean option to allow the user to prevent the automatic indexing of the cross-referencing keys see, seealso and alias.

```
251 \define@boolkey{glossaries-extra.sty}{@glxtr@}{autoseeindex}[true]{%
252 }
253 \@glxtr@autoseeindextrue
```

iesExtraWarning Allow users to suppress warnings.

```

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

```

raWarningNoLine Allow users to suppress warnings.

```

255 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
256   \PackageWarningNoLine{glossaries-extra}{#1}}

257 \@glxtr@declareoption{nowarn}{%
258   \let\GlossariesExtraWarning\@gobble
259   \let\GlossariesExtraWarningNoLine\@gobble
260   \glxtr@doption{nowarn}%
261 }

postdot Shortcut for nopostdot=false
262 \@glxtr@declareoption{postdot}{%
263   \glxtr@doption{nopostdot=false}%
264 }

glxtrabbrvtype Glossary type for abbreviations.
265 \newcommand*{\glxtrabbrvtype}{\glsdefaulttype}

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

bbreviationsdef
267 \newcommand*{\@glxtr@doabbreviationsdef}{%
268   \@ifpackageloaded{babel}{%
269     {\providecommand{\abbreviationsname}{\acronymname}}%
270     {\providecommand{\abbreviationsname}{Abbreviations}}%
271     \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
272     \renewcommand*{\glxtrabbrvtype}{abbreviations}%
273     \newcommand*{\printabbreviations}[1][1]{%
274       \printglossary[type=\glxtrabbrvtype,##1]%
275     }%
276     \disable@keys{glossaries-extra.sty}{abbreviations}%

    If the acronym option hasn't been used, change \acronymtype to \glxtrabbrvtype.
277     \ifglsacronym
278     \else
279       \renewcommand*{\acronymtype}{\glxtrabbrvtype}%
280     \fi
281 }%

abbreviations If abbreviations, create a new glossary type for abbreviations.
282 \@glxtr@declareoption{abbreviations}{%
283   \let\@glxtr@abbreviationsdef\@glxtr@doabbreviationsdef
284 }

```

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

```

285 \newcommand*\GlsXtrDefineAbbreviationShortcuts{%
286   \newcommand*\ab{\cgl}%
287   \newcommand*\abp{\cglspl}%
288   \newcommand*\as{\glxtrshort}%
289   \newcommand*\asp{\glxtrshortpl}%
290   \newcommand*\al{\glxtrlong}%
291   \newcommand*\alp{\glxtrlongpl}%
292   \newcommand*\af{\glxtrfull}%
293   \newcommand*\afp{\glxtrfullpl}%
294   \newcommand*\Ab{\cGls}%
295   \newcommand*\Abp{\cGlspl}%
296   \newcommand*\As{\Glsxtrshort}%
297   \newcommand*\Asp{\Glsxtrshortpl}%
298   \newcommand*\Al{\Glsxtrlong}%
299   \newcommand*\Alp{\Glsxtrlongpl}%
300   \newcommand*\Af{\Glsxtrfull}%
301   \newcommand*\Afp{\Glsxtrfullpl}%
302   \newcommand*\AB{\cGLS}%
303   \newcommand*\ABP{\cGLSpl}%
304   \newcommand*\AS{\GLSxtrshort}%
305   \newcommand*\ASP{\GLSxtrshortpl}%
306   \newcommand*\AL{\GLSxtrlong}%
307   \newcommand*\ALP{\GLSxtrlongpl}%
308   \newcommand*\AF{\GLSxtrfull}%
309   \newcommand*\AFP{\GLSxtrfullpl}%
310   \newcommand*\newabbr{\newabbreviation}%

```

Disable this command after it's been used.

```

311 \let\GlsXtrDefineAbbreviationShortcuts\relax
312 }

```

`\GlsXtrDefineAcShortcuts` Enable shortcut commands for the abbreviations, but uses the analogous commands provided by glossaries.

```

313 \newcommand*\GlsXtrDefineAcShortcuts{%
314   \newcommand*\ac{\cgl}%
315   \newcommand*\acp{\cglspl}%
316   \newcommand*\acs{\glxtrshort}%
317   \newcommand*\acsp{\glxtrshortpl}%
318   \newcommand*\acl{\glxtrlong}%
319   \newcommand*\aclp{\glxtrlongpl}%
320   \newcommand*\acf{\glxtrfull}%
321   \newcommand*\acfp{\glxtrfullpl}%
322   \newcommand*\Ac{\cGls}%
323   \newcommand*\Acp{\cGlspl}%
324   \newcommand*\Acs{\Glsxtrshort}%
325   \newcommand*\Acsp{\Glsxtrshortpl}%
326   \newcommand*\Acl{\Glsxtrlong}%

```

```

327 \newcommand*{\Ac1p}{\Glsxtrlongpl}%
328 \newcommand*{\Acf}{\Glsxtrfull}%
329 \newcommand*{\Acfp}{\Glsxtrfullpl}%
330 \newcommand*{\AC}{\cGLS}%
331 \newcommand*{\ACP}{\cGLSp1}%
332 \newcommand*{\ACS}{\GLSxtrshort}%
333 \newcommand*{\ACSP}{\GLSxtrshortpl}%
334 \newcommand*{\ACL}{\GLSxtrlong}%
335 \newcommand*{\ACLP}{\GLSxtrlongpl}%
336 \newcommand*{\ACF}{\GLSxtrfull}%
337 \newcommand*{\ACFP}{\GLSxtrfullpl}%
338 \newcommand*{\newabbr}{\newabbreviation}%

```

Disable this command after it's been used.

```

339 \let\GlsXtrDefineAcShortcuts\relax
340 }

```

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

```

341 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
342 \newcommand*{\newentry}{\newglossaryentry}%
343 \ifdef\printsymbols
344 {%
345 \newcommand*{\newsym}{\glxtrnewsymbol}%
346 }{}%
347 \ifdef\printnumbers
348 {%
349 \newcommand*{\newnum}{\glxtrnewnumber}%
350 }{}%
351 \let\GlsXtrDefineOtherShortcuts\relax
352 }

```

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.

```

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

```

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

```

354 \newcommand*{\@glxtr@shortcutsval}{\ifglssacrshortcuts acro\else none\fi}%

```

Provide shortcuts option. Unlike the glossaries version, this is a choice rather than a boolean key but it also provides `shortcuts=true` and `shortcuts=false`, which are equivalent to `shortcuts=all` and `shortcuts=none`. Multiple use of this option in the *same* option list will override each other. New to v1.17: `shortcuts=ac` which implements `\GlsXtrDefineAcShortcuts` (not included in `shortcuts=all` as it conflicts with other shortcuts).

```

355 \define@choicekey{glossaries-extra.sty}{shortcuts}[\val\nr]%
356 {acronyms,acro,abbreviations,abbr,other,all,true,none,false,ac}[true]{%

```

```

357 \let\@glxtr@shortcutsval\val
358 \ifcase\nr\relax % acronyms
359   \renewcommand*{\@glxtr@setupshortcuts}{%
360     \glsacrshortcutstrue
361     \DefineAcronymSynonyms
362   }%
363 \or % acro
364   \renewcommand*{\@glxtr@setupshortcuts}{%
365     \glsacrshortcutstrue
366     \DefineAcronymSynonyms
367   }%
368 \or % abbreviations
369   \renewcommand*{\@glxtr@setupshortcuts}{%
370     \GlsXtrDefineAbbreviationShortcuts
371   }%
372 \or % abbr
373   \renewcommand*{\@glxtr@setupshortcuts}{%
374     \GlsXtrDefineAbbreviationShortcuts
375   }%
376 \or % other
377   \renewcommand*{\@glxtr@setupshortcuts}{%
378     \GlsXtrDefineOtherShortcuts
379   }%
380 \or % all
381   \renewcommand*{\@glxtr@setupshortcuts}{%
382     \glsacrshortcutstrue
383     \DefineAcronymSynonyms
384     \GlsXtrDefineAbbreviationShortcuts
385     \GlsXtrDefineOtherShortcuts
386   }%
387 \or % true
388   \renewcommand*{\@glxtr@setupshortcuts}{%
389     \glsacrshortcutstrue
390     \DefineAcronymSynonyms
391     \GlsXtrDefineAbbreviationShortcuts
392     \GlsXtrDefineOtherShortcuts
393   }%
394 \or % none, false
395   \renewcommand*{\@glxtr@setupshortcuts}{}%
396 \or % ac
397   \renewcommand*{\@glxtr@setupshortcuts}{%
398     \glsacrshortcutstrue
399     \GlsXtrDefineAcShortcuts
400   }%
401 \fi
402 }

```

lsxtr@doaccsupp

```

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

```


accsupp If accsupp, load glossaries-accsupp package.

```
404 \@glsxtr@declareoption{accsupp}{%  
405 \renewcommand*{\@glsxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}
```

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

```
406 \newcommand{\glsxtrNoGlossaryWarning}[1]{%  
407 \@glsxtr@defaultnoglossarywarning{#1}%  
408 }
```

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

```
409 \define@choicekey{glossaries-extra.sty}{nomissingglstext}[\val\nr]%  
410 {true,false}[true]{%  
411 \ifcase\nr\relax % true  
412 \renewcommand{\glsxtrNoGlossaryWarning}[1]{%  
413 \null  
414 }%  
415 \else % false  
416 \renewcommand{\glsxtrNoGlossaryWarning}[1]{%  
417 \@glsxtr@defaultnoglossarywarning{#1}%  
418 }%  
419 \fi  
420 }
```

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

xtr@redefstyles

```
421 \newcommand*{\@glsxtr@redefstyles}{}
```

stylemods

```
422 \define@key{glossaries-extra.sty}{stylemods}{%  
423 \ifblank{#1}%  
424 {%  
425 \renewcommand*{\@glsxtr@redefstyles}{%  
426 \RequirePackage{glossaries-extra-stylemods}}%  
427 }%  
428 {%  
429 \renewcommand*{\@glsxtr@redefstyles}{}%  
430 \@for\@glsxtr@tmp:=#1\do{%  
431 \IfFileExists{glossary-\@glsxtr@tmp.sty}%  
432 {%  
433 \eappto\@glsxtr@redefstyles{%  
434 \noexpand\RequirePackage{glossary-\@glsxtr@tmp}}%  
435 }%  
436 {%  
437 \PackageError{glossaries-extra}%  
438 {Glossaries style package ‘glossary-\@glsxtr@tmp.sty’  
439 doesn’t exist (did you mean to use the ‘style’ key?)}%  
440 {The list of values (#{1}) in the ‘stylemods’ key should
```

```

441             match the glossary-xxx.sty files provided with
442             glossaries.sty}%
443     }%
444 }%
445 \appto\@glsxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
446 }%
447 }

glsxtr@do@style
448 \newcommand*{\@glsxtr@do@style}{%

style Since the stylemods option can automatically load extra style packages, deal with the style
option after those packages have been loaded.
449 \define@key{glossaries-extra.sty}{style}{%
450 \renewcommand*{\@glsxtr@do@style}{%
Set this as the default style:
451 \setkeys{glossaries.sty}{style={#1}}%
Set this style:
452 \setglossarystyle{#1}%
453 }%
454 }

Pass all other options to glossaries.
455 \DeclareOptionX*{%
456 \expandafter\glsxtr@doooption\expandafter{\CurrentOption}}

Process options.
457 \ProcessOptionsX

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

Load the glossaries-accsupp package if required.
459 \@glsxtr@doaccsupp

g@doseeglossary Save original definition of \@do@seeglossary
460 \let\@glsxtr@org@doseeglossary\@do@seeglossary

@org@gloautosee Save and restore original definition of \@glo@autosee. (That command may not be defined
as it was only introduced to glossaries v4.30, in which case the synonym won't be defined
either.)
461 \let\@glsxtr@org@gloautosee\@glo@autosee

Check if user tried autoseeindex=false when it can't be supported.
462 \if@glsxtr@autoseeindex
463 \else
464 \ifdef\@glsxtr@org@gloautosee
465 {}%
```

```

466 {\PackageError{glossaries-extra}{‘autoseeindex=false’ package
467 option requires at least v4.30 of glossaries.sty}%
468 {You need to update the glossaries.sty package}%
469 }
470 \fi

\@glo@autosee If \@glo@autosee has been defined (glossaries v4.30 onwards), redefine it to test the au-
toseeindex option.
471 \ifdef\@glo@autosee
472 {%
473 \renewcommand*{\@glo@autosee}{%
474 \if@glxtr@autoseeindex\@glxtr@org@glo@autosee\fi}%
475 }%
476 {}

checkseeallowed Don't prohibit the use of the see key before the indexing files have been opened if the auto-
matic see indexing has been disabled, since it's no longer an issue.
477 \renewcommand*{\gls@checkseeallowed}{%
478 \if@glxtr@autoseeindex\@gls@see@noindex\fi
479 }

Define abbreviations glossaries if required.
480 \@glxtr@abbreviationsdef
481 \let\@glxtr@abbreviationsdef\relax

Setup shortcuts if required.
482 \@glxtr@setupshortcuts

Redefine \@glxtr@redef@for@gl@sentries if required.
483 \@glxtr@redef@for@gl@sentries

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

Now define the user command:
485 \newcommand*{\glossariesextrasetup}[1]{%
486 \let\glxtr@setup@record\relax
487 \let\@glxtr@setupshortcuts\relax
488 \let\@glxtr@redef@for@gl@sentries\relax
489 \setkeys{glossaries-extra.sty}{#1}%
490 \@glxtr@abbreviationsdef
491 \let\@glxtr@abbreviationsdef\relax
492 \@glxtr@setupshortcuts
493 \glxtr@setup@record
494 \@glxtr@redef@for@gl@sentries
495 }

@@do@wrglossary Save original definition of @@do@wrglossary.
496 \let\glxtr@@do@wrglossary\@@do@wrglossary

```

saveentrycounter Save original definition of \@gls@saveentrycounter.
497 \let\glstr@saveentrycounter\@gls@saveentrycounter

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

Set up record option if required.
499 \glstr@setup@record

Disable preamble-only options and switch on the undefined tag at the start of the document.
500 \AtBeginDocument{%
501 \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
502 \def\@glstrundeftag{\glstrundeftag}%
503 }

1.2 Extra Utilities

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

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

```
504 \newcommand{\glstrifemptyglossary}[3]{%
505   \ifcsdef{glolist@#1}%
506   {%
507     \ifcsstring{glolist@#1}{,}{#2}{#3}%
508   }%
509   {%
510     \glstrundefaction{Glossary type '#1' doesn't exist}{}%
511     #2%
512   }%
513 }
```

trifkeydefined Tests if the key given in the first argument has been defined.
514 \newcommand*\glstrifkeydefined[3]{%
515 \key@ifundefined{glossentry}{#1}{#3}{#2}%
516 }

providestoragekey Like `\glsaddstoragekey` but does nothing if the key has already been defined.
517 \newcommand*\glstrprovidestoragekey{%

```

518 \ifstar\@glsxtr@provide@storagekey\@glsxtr@provide@storagekey
519 }

```

vide@storagekey Unstarred version.

```

520 \newcommand*{\@glsxtr@provide@storagekey}[3]{%
521   \key@ifundefined{glossentry}{#1}%
522   {%
523     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
524     \appto\@gls@keymap{,{#1}{#1}}%
525     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
526     \appto\@newglossaryentryposthook{%
527       \letcs{\@glo@tmp}{@glo@#1}%
528       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
529     }%

```

Allow the user to omit the user level command if they only intended fetching the value with
\glsxtrusefield

```

530   \ifblank{#3}
531   {}%
532   {%
533     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
534   }%
535 }%
536 {%

```

Provide the no-link command if not already defined.

```

537   \ifblank{#3}
538   {}%
539   {%
540     \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
541   }%
542 }%
543 }

```

vide@storagekey Starred version.

```

544 \newcommand*{\s@glsxtr@provide@storagekey}[1]{%
545   \key@ifundefined{glossentry}{#1}%
546   {%
547     \expandafter\newcommand\expandafter*\expandafter
548     {\csname gls@assign@#1@field\endcsname}[2]{%
549       \@gls@expand@field{##1}{#1}{##2}%
550     }%
551   }%
552   {}%
553   \@glsxtr@provide@addstoragekey{#1}%
554 }

```

The name of a text-block control sequence can be stored in a field (given by \GlsXtrFmtField).
This command can then be used with \glsxtrfmt [*options*] {<label>} {<text>} which effec-

tively does `\glslink[⟨options⟩]{⟨label⟩}{⟨cs⟩{⟨text⟩}}` If the field hasn't been set for that entry just `⟨text⟩` is done.

`\GlsXtrFmtField`

```
555 \newcommand{\GlsXtrFmtField}{useri}
```

`tDefaultOptions`

```
556 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}
```

`\glsxtrfmt` The post-link hook isn't done.

```
557 \newrobustcmd*{\glsxtrfmt}[3] [] {%
558   \glsdoifexistsordo{#2}%
559   {%
560     \ifglshasfield{\GlsXtrFmtField}{#2}%
561     {%
562       \let\do@glsl@link@checkfirsthyper\relax
563       \expandafter\@glsl@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
564       {\csuse{\glscurrentfieldvalue}{#3}}%
565     }%
566     {#3}%
567   }%
568   {#3}%
569 }
```

`\glstxentryfmt` No link or indexing.

```
570 \ifdef\texorpdfstring
571 {
572   \newcommand*{\glstxentryfmt}[2] {%
573     \texorpdfstring{\@glstxentryfmt{#1}{#2}}{#2}%
574   }
575 }
576 {
577   \newcommand*{\glstxentryfmt}{\@glstxentryfmt}
578 }
```

`@glstxentryfmt`

```
579 \newrobustcmd*{\@glstxentryfmt}[2] {%
580   \glsdoifexistsordo
581   {%
582     \ifglshasfield{\GlsXtrFmtField}{#1}%
583     {%
584       \csuse{\glscurrentfieldvalue}{#2}%
585     }%
586     {#2}%
587   }%
588   {#2}%
589 }
```

`trfieldlistadd` If a field stores an etoolbox internal list (e.g. `loclist`) then this macro provides a convenient way of adding to the list via etoolbox's `\listcsadd`. The first argument is the entry's label, the second is the field label and the third is the element to add to the list.

```
590 \newcommand*{\glstrfieldlistadd}[3]{%
591   \listcsadd{glo@\glstetoklabel{#1}@#2}{#3}%
592 }
```

`trfieldlistgadd` Similarly but uses `\listcsgadd`.

```
593 \newcommand*{\glstrfieldlistgadd}[3]{%
594   \listcsgadd{glo@\glstetoklabel{#1}@#2}{#3}%
595 }
```

`trfieldlisteadd` Similarly but uses `\listcseadd`.

```
596 \newcommand*{\glstrfieldlisteadd}[3]{%
597   \listcseadd{glo@\glstetoklabel{#1}@#2}{#3}%
598 }
```

`trfieldlistxadd` Similarly but uses `\listcsxadd`.

```
599 \newcommand*{\glstrfieldlistxadd}[3]{%
600   \listcsxadd{glo@\glstetoklabel{#1}@#2}{#3}%
601 }
```

Now provide commands to iterate over these lists.

`fielddolistloop`

```
602 \newcommand*{\glstrfielddolistloop}[2]{%
603   \dolistcsloop{glo@\glstetoklabel{#1}@#2}%
604 }
```

`fieldforlistloop`

```
605 \newcommand*{\glstrfieldforlistloop}[3]{%
606   \forlistcsloop{glo@\glstetoklabel{#1}@#2}{#3}%
607 }
```

List element tests:

`trfieldifinlist` First argument label, second argument field, third argument item, fourth true part and fifth false part.

```
608 \newcommand*{\glstrfieldifinlist}[5]{%
609   \ifinlistcs{#3}{glo@\glstetoklabel{#1}@#2}{#4}{#5}%
610 }
```

`trfieldxifinlist` Expands item.

```
611 \newcommand*{\glstrfieldxifinlist}[5]{%
612   \xifinlistcs{#3}{glo@\glstetoklabel{#1}@#2}{#4}{#5}%
613 }
```

`\glxtrusefield` Provide a user-level alternative to `\@gls@entry@field`. The first argument is the entry label. The second argument is the field label.

```

614 \newcommand*{\glxtrusefield}[2]{%
615   \@gls@entry@field{#1}{#2}%
616 }

```

`\Glsxtrusefield` Provide a user-level alternative to `\@Gls@entry@field`.

```

617 \newcommand*{\Glsxtrusefield}[2]{%
618   \@gls@entry@field{#1}{#2}%
619 }

```

`\glxtrdeffield` Just use `\csdef` to provide a field value for the given entry.

```

620 \newcommand*{\glxtrdeffield}[2]{\csdef{glo@\glsdetoklabel{#1}@#2}}

```

`glxtredeffield` Just use `\csedef` to provide a field value for the given entry.

```

621 \newcommand*{\glxtredeffield}[2]{\csedef{glo@\glsdetoklabel{#1}@#2}}

```

`etfieldifexists`

```

622 \newcommand*{\glxtrsetfieldifexists}[3]{\glsdoidexists{#1}{#3}}

```

`\GlsXtrSetField` Allow the user to set a field. First argument entry label, second argument field label, third argument value.

```

623 \newrobustcmd*{\GlsXtrSetField}[3]{%
624   \glxtrsetfieldifexists{#1}{#2}%
625   {\csdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
626 }

```

`\GlsXtrLetField` Uses `\cslet` instead. Third argument should be a macro.

```

627 \newrobustcmd*{\GlsXtrLetField}[3]{%
628   \glxtrsetfieldifexists{#1}{#2}%
629   {\cslet{glo@\glsdetoklabel{#1}@#2}{#3}}%
630 }

```

`sGlsXtrLetField` Uses `\csletcs` instead. Third argument should be a control sequence name.

```

631 \newrobustcmd*{\csGlsXtrLetField}[3]{%
632   \glxtrsetfieldifexists{#1}{#2}%
633   {\csletcs{glo@\glsdetoklabel{#1}@#2}{#3}}%
634 }

```

`LetFieldToField` Sets the field for one entry to the field for another entry. Third argument should be the other entry and the fourth argument that other field label.

```

635 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
636   \glxtrsetfieldifexists{#1}{#2}%
637   {\csletcs{glo@\glsdetoklabel{#1}@#2}{glo@\glsdetoklabel{#3}@#4}}%
638 }

```


`gGlsXtrSetField` Allow the user to set a field. First argument entry label, second argument field label, third argument value.

```
639 \newrobustcmd*{\gGlsXtrSetField}[3]{%
640   \glsxtrsetfieldifexists{#1}{#2}%
641   {\csgdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
642 }
```

`xGlsXtrSetField`

```
643 \newrobustcmd*{\xGlsXtrSetField}[3]{%
644   \glsxtrsetfieldifexists{#1}{#2}%
645   {\protected@csxdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
646 }
```

`eGlsXtrSetField`

```
647 \newrobustcmd*{\eGlsXtrSetField}[3]{%
648   \glsxtrsetfieldifexists{#1}{#2}%
649   {\protected@csedef{glo@\glsdetoklabel{#1}@#2}{#3}}%
650 }
```

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

```
651 \ifglsentrycounter
652   \newcommand*{\glsxtrpageref}[1]{\pageref{glsentry-\glsdetoklabel{#1}}}
653 \else
654   \ifglsesubentrycounter
655     \newcommand*{\glsxtrpageref}[1]{\pageref{glsentry-\glsdetoklabel{#1}}}
656   \else
657     \newcommand*{\glsxtrpageref}[1]{\gls{#1}}
658   \fi
659 \fi
```

`lossarypreamble`

```
660 \newcommand{\apptoglossarypreamble}[2][\glsdefaulttype]{%
661   \ifcsdef{glolist@#1}%
662   {%
663     \ifcsundef{@glossarypreamble@#1}%
664     {\csdef{@glossarypreamble@#1}{}}%
665     {}%
666     \csappto{@glossarypreamble@#1}{#2}%
667   }%
668   {%
669     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
670   }%
671 }
```

`lossarypreamble`

```
672 \newcommand{\preglossarypreamble}[2][\glsdefaulttype]{%
673   \ifcsdef{glolist@#1}%
674   {%
```

```

675 \ifcsundef{@glossarypreamble@#1}%
676 {\csdef{@glossarypreamble@#1}{}}%
677 {}%
678 \cspretto{@glossarypreamble@#1}{#2}%
679 }%
680 {%
681 \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
682 }%
683 }

```

1.3 Modifications to Commands Provided by glossaries

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

Provide a starred version of `\longnewglossaryentry` that doesn't automatically insert `\leavevmode\unskip\nopostdesc` at the end of the description. The unstarred version is modified to use `\glxtrpostlongdescription` instead.

`ewglossaryentry`

```

684 \renewcommand*{\longnewglossaryentry}{%
685 \@ifstar{\glxtr@s@longnewglossaryentry\@glxtr@longnewglossaryentry
686 }

```

`ewglossaryentry`

Starred version.

```

687 \newcommand{\@glxtr@s@longnewglossaryentry}[3]{%
688 \glsdoifnoexists{#1}%
689 {%
690 \bgroup
691 \let\@org@newglossaryentryprehook\@newglossaryentryprehook
692 \long\def\@newglossaryentryprehook{%
693 \long\def\@glo@desc{#3}%
694 \@org@newglossaryentryprehook
695 }%
696 \renewcommand*{\gls@assign@desc}[1]{%
697 \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
698 \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
699 }
700 \gls@defglossaryentry{#1}{#2}%
701 \egroup
702 }%
703 }

```

`ewglossaryentry`

Unstarred version.

```

704 \newcommand{\@glxtr@longnewglossaryentry}[3]{%
705 \glsdoifnoexists{#1}%
706 {%
707 \bgroup

```

```

708 \let\@org@newglossaryentryprehook\@newglossaryentryprehook
709 \long\def\@newglossaryentryprehook{%
710 \long\def\@glo@desc{#3\glstrpostlongdescription}%
711 \@org@newglossaryentryprehook
712 }%
713 \renewcommand*{\gls@assign@desc}[1]{%
714 \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%

```

The following is different from the base glossaries.sty:

```

715 \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
716 }
717 \gls@defglossaryentry{#1}{#2}%
718 \egroup
719 }%
720 }

```

`\longdescription` Hook at the end of the description when using the unstarred `\longnewglossaryentry`.

```

721 \newcommand*{\glstrpostlongdescription}{\leavevmode\unskip\nopostdesc}

```

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

`\ignoredglossary` Redefine to check for star.

```

722 \renewcommand{\newignoredglossary}{%
723 \@ifstar\glstr@s@newignoredglossary\glstr@org@newignoredglossary
724 }

```

`\ignoredglossary` The original definition is patched to check for existence.

```

725 \newcommand*{\glstr@org@newignoredglossary}[1]{%
726 \ifcsdef{glolist@#1}
727 {%
728 \glstrundefaction{Glossary type ‘#1’ already exists}{}%
729 }%
730 {%
731 \ifdefempty\@ignored@glossaries
732 {%
733 \edef\@ignored@glossaries{#1}%
734 }%
735 {%
736 \eappto\@ignored@glossaries{,#1}%
737 }%
738 \csgdef{glolist@#1}{,}%
739 \ifcsundef{gls@#1@entryfmt}%
740 {%
741 \defglsentryfmt[#1]{\glsentryfmt}%
742 }%
743 }%
744 \ifdefempty\@gls@nohyperlist
745 {%

```

```

746     \renewcommand*{\@gls@nohyperlist}{#1}%
747   }%
748   {%
749     \eappto\@gls@nohyperlist{, #1}%
750   }%
751 }%
752 }

```

ignoredglossary Starred form.

```

753 \newcommand*{\glsxtr@s@newignoredglossary}[1]{%
754   \ifcsdef{glolist@#1}
755   {%
756     \glsxtrundefaction{Glossary type ‘#1’ already exists}{}%
757   }%
758   {%
759     \ifdefempty\@ignored@glossaries
760     {%
761       \edef\@ignored@glossaries{#1}%
762     }%
763     {%
764       \eappto\@ignored@glossaries{, #1}%
765     }%
766     \csgdef{glolist@#1}{,}%
767     \ifcsundef{gls@#1@entryfmt}%
768     {%
769       \defglsentryfmt[#1]{\glsentryfmt}%
770     }%
771     {}%
772   }%
773 }

```

\glssettocitle Ignored glossaries don’t have an associated title, so modify \glssettocitle to check for it to prevent an undefined command written to the toc file.

```

774 \glsifusetranslator
775 {%
776   \renewcommand*{\glssettocitle}[1]{%
777     \ifcsdef{gls@tr@set@#1@tocitle}%
778     {%
779       \csuse{gls@tr@set@#1@tocitle}%
780     }%
781     {%
782       \ifcsdef{@glotype@#1@title}%
783       {\def\glossarytocitle{\csname @glotype@#1@title\endcsname}}%
784       {\def\glossarytocitle{\glossarytitle}}%
785     }%
786   }%
787 }
788 {
789   \renewcommand*{\glssettocitle}[1]{%

```

```

790 \ifcsdef{@glotype@#1@title}%
791 {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
792 {\def\glossarytoctitle{\glossarytitle}}%
793 }
794 }

```

ignoredglossary As above but won't do anything if the glossary already exists.

```

795 \newcommand{\provideignoredglossary}{%
796 \@ifstar\glxtr@s@provideignoredglossary\glxtr@provideignoredglossary
797 }

```

ignoredglossary Unstarred version.

```

798 \newcommand*{\glxtr@provideignoredglossary}[1]{%
799 \ifcsdef{glolist@#1}
800 {}%
801 {%
802 \ifdefempty\@ignored@glossaries
803 {%
804 \edef\@ignored@glossaries{#1}%
805 }%
806 {%
807 \eappto\@ignored@glossaries{,#1}%
808 }%
809 \csgdef{glolist@#1}{,}%
810 \ifcsundef{gls@#1@entryfmt}%
811 {%
812 \def\glsentryfmt[#1]{\glsentryfmt}%
813 }%
814 {}%
815 \ifdefempty\@gls@nohyperlist
816 {%
817 \renewcommand*{\@gls@nohyperlist}{#1}%
818 }%
819 {%
820 \eappto\@gls@nohyperlist{,#1}%
821 }%
822 }%
823 }

```

ignoredglossary Starred form.

```

824 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
825 \ifcsdef{glolist@#1}
826 {}%
827 {%
828 \ifdefempty\@ignored@glossaries
829 {%
830 \edef\@ignored@glossaries{#1}%
831 }%
832 {%

```

```

833     \eappto\@ignored@glossaries{,#1}%
834   }%
835   \csgdef{glolist@#1}{,}%
836   \ifcsundef{gls@#1@entryfmt}%
837   {%
838     \defglsentryfmt[#1]{\glsentryfmt}%
839   }%
840   {}%
841 }%
842 }

```

`\xtrcopytoglossary` Adds an entry label to another glossary list. First argument is entry label. Second argument is glossary label.

```

843 \newcommand*\xtrcopytoglossary[2]{%
844   \glsdoifexists{#1}%
845   {%
846     \ifcsdef{glolist@#2}
847     {%
848       \cseappto{glolist@#2}{#1,}%
849     }%
850   }%
851   \glsxtrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
852 }%
853 }%
854 }

```

1.3.1 Existence Checks

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

```

855 \renewcommand*\glsdoifexists[2]{%
856   \ifglsentryexists{#1}{#2}%
857   {%

```

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

```

858   \edef\glslabel{\glsdetoklabel{#1}}%
859   \glsxtrundefaction{Glossary entry ‘\glslabel’
860     has not been defined}{You need to define a glossary entry before
861     you can reference it.}%
862   }%
863 }

```

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

```

864 \renewcommand*\glsdoifnoexists[2]{%
865   \ifglsentryexists{#1}{%
866     \glsxtrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
867       has already been defined}{}}{#2}%
868 }

```

`\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.

```

869 \ifdef\glsdoifexistsordo
870 {%
871   \renewcommand{\glsdoifexistsordo}[3]{%
872     \ifglstryexists{#1}{#2}%
873     {%
874       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
875       has not been defined}{You need to define a glossary entry
876       before you can use it.}%
877       #3%
878     }%
879   }%
880 }
881 {%
882   \glstr@warnonexistsordo\glsdoifexistsordo
883   \newcommand{\glsdoifexistsordo}[3]{%
884     \ifglstryexists{#1}{#2}%
885     {%
886       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
887       has not been defined}{You need to define a glossary entry
888       before you can use it.}%
889       #3%
890     }%
891   }%
892 }

```

`\glsarynoexistsordo` Similarly for `\doifglossarynoexistsordo`.

```

893 \ifdef\doifglossarynoexistsordo
894 {%
895   \renewcommand{\doifglossarynoexistsordo}[3]{%
896     \ifglossaryexists{#1}%
897     {%
898       \glstrundefaction{Glossary type '#1' already exists}{}%
899       #3%
900     }%
901     {#2}%
902   }%
903 }
904 {%
905   \glstr@warnonexistsordo\doifglossarynoexistsordo
906   \newcommand{\doifglossarynoexistsordo}[3]{%
907     \ifglossaryexists{#1}%
908     {%
909       \glstrundefaction{Glossary type '#1' already exists}{}%
910       #3%
911     }%
912     {#2}%
913   }%

```

```

914 }
915

```

There are now three types of cross-references: the see key (as original), the alias key (from glossaries-extra v1.12) and the seealso key (from glossaries-extra v1.16). The original see key needs to have a corresponding field (which it doesn't with the base glossaries package).

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

```

916 \appto\@newglossaryentryposthook{%
917   \ifdefvoid\@glo@see
918     {\csxdef{glo@\@glo@label @see}{}}%
919     {%
920       \csxdef{glo@\@glo@label @see}{\@glo@see}%
921       \ifglstr@autoseeindex
922         \@glstr@autoindexcrossrefs
923       \fi
924     }%
925 }
926 \appto\@gls@keymap{,{see}{see}}

```

`\glstrusee` Apply `\glseeformat` to the see key if not empty.

```

927 \newcommand*{\glstrusee}[1]{%
928   \glsoifexists{#1}%
929   {%
930     \letcs{\@glo@see}{glo@\glstetoklabel{#1}@see}%
931     \ifdefempty\@glo@see
932       {}%
933       {%
934         \expandafter\glstr@usee\@glo@see\end@glstr@usee
935       }%
936     }%
937 }

```

`\glstr@usee`

```

938 \newcommand*{\glstr@usee}[1][\seename]{%
939   \@glstr@usee[#1]%
940 }

```

`\@glstr@usee`

```

941 \def\@glstr@usee[#1]#2\end@glstr@usee{%
942   \glstruseeformat{#1}{#2}%
943 }

```

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

```

944 \newcommand*{\glstruseeformat}[2]{%
945   \glseeformat{#1}{#2}{}%
946 }

```


`\glxtruseseealso` Apply `\glseeformat` to the `seealso` key if not empty. There's no optional tag to worry about here.

```

947 \newcommand*{\glxtruseseealso}[1]{%
948   \glsoifexists{#1}%
949   {%
950     \letcs{\@glo@see}{glo\glstdetoklabel{#1}@seealso}%
951     \ifdefempty\@glo@see
952     {}%
953     {%
954       \expandafter\glxtruseseealsoformat\expandafter{\@glo@see}%
955     }%
956   }%
957 }

```

`\glseealsoformat` The format used by `\glxtruseseealso`. The argument is the comma-separated list of cross-referenced labels.

```

958 \newcommand*{\glxtruseseealsoformat}[1]{%
959   \glseeformat[\seealso name]{#1}%
960 }

```

`\glxtrseelist` Fully expands argument before passing to `\glseeelist`. (The argument to `\glseeelist` must be a comma-separated list of entry labels.)

```

961 \newrobustcmd{\glxtrseelist}[1]{%
962   \edef\@glo@tmp{\noexpand\glseeelist{#1}}\@glo@tmp
963 }

```

`\seealso name` In case this command hasn't been defined. (Should be provided by language packages.)

```

964 \providecommand{\seealso name}{see also}

```

`\glxtrindexseealso` If `\@xdycrossrefhook` is defined, provide a `seealso` crossref class. Otherwise this just does `\glsee` with `\seealso name` as the tag. The hook is only defined if both `xindy` and `glossaries v4.30+` are being used.

```

965 \ifdef\@xdycrossrefhook
966 {
  Add the cross-reference class definition to the hook.
967   \appto\@xdycrossrefhook{%
968     \write\glswrite{(define-crossref-class \string"seealso\string"
969       :unverified )}%
970     \write\glswrite{(markup-crossref-list
971       :class \string"seealso\string"^^J\space\space\space
972       :open \string"\string\glxtruseseealsoformat\glsoopenbrace\string"
973       :close \string"\glsclosebrace\string")}%
974   }

```

Append to class list.

```

975   \appto\@xdylocationclassorder{\space\string"seealso\string"}

```

This essentially works like `\@do@seeglossary` but uses the `seealso` class.

```

976 \newrobustcmd*{\glxtrindexseealso}[2]{%
977   \def\@gls@xref{#2}%
978   \@onelevel@sanitize\@gls@xref
979   \@gls@checkmkidxchars\@gls@xref
980   \gls@glossary{\csname glo@#1@type\endcsname}{%
981     (indexentry
982       :tkey (\csname glo@#1@index\endcsname)
983       :xref (\string"\@gls@xref\string")
984       :attr \string"seealso\string"
985     )
986   }%
987 }
988 }
989 {

```

`xindy` not in use or `glossaries` version too old to support this.

```

990 \newrobustcmd*{\glxtrindexseealso}{\glssee[\seealso]}
991 }

```

The `alias` key should be set to the label of the synonymous entry. The `seealso` key essentially behaves like `see=[\seealso]{\<xr-list>}`. Neither of these new keys has the optional tag part allowed with `see`.

If `\gls@set@xr@key` has been defined (`glossaries v4.30`), use that, otherwise just use `\glsaddstoragekey`.

```

992 \ifdef\gls@set@xr@key
993 {

```

We have at least `glossaries v4.30`. This means the new keys can be governed by the same settings as the `see` key.

```

994 \define@key{glossentry}{alias}{%
995   \gls@set@xr@key{alias}{\@glo@alias}{#1}%
996 }
997 \define@key{glossentry}{seealso}{%
998   \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
999 }

```

Add to the key mappings.

```

1000 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}

```

Set the default value.

```

1001 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%

```

Assign the field values.

```

1002 \appto\@newglossaryentryposthook{%
1003   \ifdefvoid\@glo@seealso
1004     {\csxdef{glo@\@glo@label @seealso}{}}%
1005     {%
1006       \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1007       \if@glxtr@autoseeindex

```

```

1008      \@glsxtr@autoindexcrossrefs
1009      \fi
1010    }%

```

The alias field doesn't trigger the automatic cross-reference indexing performed at the end of the document.

```

1011    \ifdefvoid\@glo@alias
1012      {\csxdef{glo@\@glo@label @alias}{}}%
1013      {%
1014        \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1015      }%
1016  }

```

Provide user-level commands to access the values.

`\glsxtralias`

```

1017  \newcommand*\glsxtralias}[1]{\@gls@entry@field{#1}{alias}}

```

`trseealsolabels`

```

1018  \newcommand*\glsxtrseealsolabels}[1]{\@gls@entry@field{#1}{seealso}}

```

Add to the `\@glo@autosee` hook.

```

1019  \appto\@glo@autoseehook{%
1020    \ifdefvoid\@glo@alias
1021      {%
1022        \ifdefvoid\@glo@seealso
1023          {}%
1024          {%
1025            \edef\@do@glssee{\noexpand\glsxtrindexseealso
1026              {\@glo@label}{\@glo@seealso}}%
1027            \@do@glssee
1028          }%
1029        }%
1030      {%

```

Add cross-reference if see key hasn't been used.

```

1031    \ifdefvoid\@glo@see
1032      {%
1033        \edef\@do@glssee{\noexpand\glssee{\@glo@label}{\@glo@alias}}%
1034        \@do@glssee
1035      }%
1036      {}%
1037    }%
1038  }%
1039 }
1040 {

```

We have an older version of glossaries, so just use `\glsaddstoragekey`.

`\glsxtralias`

```

1041  \glsaddstoragekey*{alias}{}{\glsxtralias}

```

trseealsolabels

```
1042 \glsaddstoragekey*{seealso}{\glsxtrseealsolabels}
```

If `\gls@set@xr@key` isn't defined, then `\@glo@autosee` won't be either, so use the post entry definition hook.

ryentryposthook Append to the hook to check for the alias and seealso keys.

```
1043 \appto\@newglossaryentryposthook{%
1044   \ifcsvoid{glo@\@glo@label @alias}%
1045   {%
1046     \ifcsvoid{glo@\@glo@label @seealso}%
1047     {}%
1048     {%
1049       \edef\@do@glsssee{\noexpand\glsxtrindexseealso
1050         {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1051       \@do@glsssee
1052     }%
1053   }%
1054   {%
```

Add cross-reference if see key hasn't been used.

```
1055   \ifdefvoid\@glo@see
1056   {%
1057     \edef\@do@glsssee{\noexpand\glsssee
1058       {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1059     \@do@glsssee
1060   }%
1061   {}%
1062 }%
1063 }
1064 }
```

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

```
1065 \AtEndDocument{\if@glxtrindexcrossrefs\glxtraddallcrossrefs\fi}
```

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

```
1066 \newcommand*\glxtraddallcrossrefs{%
1067   \forallglossaries{\@glo@type}%
1068   {%
1069     \forglsentries[\@glo@type]{\@glo@label}%
1070     {%
1071       \ifglssused{\@glo@label}%
1072       {\expandafter\@glxtr@addunusedxrefs\expandafter{\@glo@label}}}%
1073     }%
1074   }%
1075 }
```

`@addunusedxrefs` If the given entry has a `see` or `seealso` field add all unused cross-references. (The `alias` field isn't checked.)

```

1076 \newcommand*{\@glsxtr@addunusedxrefs}[1]{%
1077   \letcs{\@glo@see}{glo@glstetoklabel{#1}@see}%
1078   \ifdefvoid\@glo@see
1079   {}%
1080   {%
1081     \expandafter\glsxtr@addunused\@glo@see\@end@glsxtr@addunused
1082   }%
1083   \letcs{\@glo@see}{glo@glstetoklabel{#1}@seealso}%
1084   \ifdefvoid\@glo@see
1085   {}%
1086   {%
1087     \expandafter\glsxtr@addunused\@glo@see\@end@glsxtr@addunused
1088   }%
1089 }

```

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

```

1090 \newcommand*{\glsxtr@addunused}[1] [] {%
1091   \@glsxtr@addunused
1092 }

```

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

```

1093 \def\@glsxtr@addunused#1\@end@glsxtr@addunused{%
1094   \@for\@glsxtr@label:=#1\do
1095   {%
1096     \ifglsused{\@glsxtr@label}{}%
1097     {%
1098       \glsadd[format=glsxtrunusedformat]{\@glsxtr@label}%
1099       \glsunset{\@glsxtr@label}%
1100       \expandafter\glsxtr@addunusedxrefs\expandafter{\@glsxtr@label}%
1101     }%
1102   }%
1103 }

```

`glsxtrunusedformat`

```

1104 \newcommand*{\glsxtrunusedformat}[1]{\unskip}

```

1.3.2 Document Definitions

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

```

1105 \let\glsxtr@orgmakenoidxglossaries\makenoidxglossaries
1106 \renewcommand{\makenoidxglossaries}{%
1107   \glsxtr@orgmakenoidxglossaries
1108   \if@glsxtrdocdefrestricted

```

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

```

1109 \renewcommand*{\@gls@reference}[3]{%
1110 \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
1111 \ifinlistcs{##2}{\glsref@##1}%
1112 }%
1113 {\listcsgadd{\glsref@##1}{##2}}%
1114 \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
1115 {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
1116 }%
1117 \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
1118 }%
1119 \else

```

Disable document definitions.

```

1120 \@glsxtrdocdeffalse
1121 \fi
1122 \disable@keys{glossaries-extra.sty}{docdef}%
1123 }

```

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

```

1124 \renewcommand*{\gls@defdocnewglossaryentry}{%
1125 \ifcase\@glsxtr@docdefval
1126 docdef=false:
1127 \renewcommand*{\newglossaryentry}[2]{%
1128 \PackageError{glossaries-extra}{Glossary entries must
1129 be \MessageBreak defined in the preamble with \MessageBreak
1130 package option 'docdef=false'\MessageBreak(consider using
1131 'docdef=restricted')}{Move your glossary definitions to
1132 the preamble. You can also put them in a \MessageBreak separate file
1133 and load them with \string\loadglsentries.}%
1134 }%
1135 \or

```

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

```

1135 \let\gls@checkseeallowed\relax
1136 \let\newglossaryentry\newglossaryentry
1137 \or

```

Restricted mode just needs to allow the `see` value.

```

1138 \let\gls@checkseeallowed\relax
1139 \fi
1140 }%

```

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

```
1141 \newcommand*{\GlsXtrEnableOnTheFly}{%
1142   \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
1143 }
```

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.)

```
1144 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
1145   \renewcommand*{\glsdetoklabel}[1]{%
1146     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
1147     {%
1148       \expandafter\detokenize\expandafter{##1}%
1149     }%
1150     {\detokenize{##1}}%
1151   }%
1152   \@GlsXtrEnableOnTheFly
1153 }
1154 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
1155   \expandafter\if\glsbackslash#1%
1156   #3%
1157   \else
1158   #4%
1159   \fi
1160 }
```

sxtrstarflywarn

```
1161 \newcommand*{\glsxtrstarflywarn}{%
1162   \GlossariesExtraWarning{Experimental starred version of
1163   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
1164   read the warnings in the glossaries-extra user manual)}%
1165 }
```

rEnableOnTheFly

```
1166 \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

```
1167   \newcommand*{\glsxtrcat}{general}
```

\glsxtr

```
1168   \newcommand*{\glsxtr}[1] [] {%
1169     \def\glsxtr@keylist{##1}%
1170     \@glsxtr
1171   }
```

```

\@glxstr
1172 \newcommand*\@glxstr}[2] [] {%
1173 \ifglentryexists{##2}%
1174 {%
1175 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
1176 }%
1177 {%
1178 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
1179 description={\nopostdesc},##1}%
1180 }%
1181 \expandafter\gls\expandafter[\glxstr@keylist]{##2}%
1182 }

\Glsxtr
1183 \newcommand*\Glsxtr}[1] [] {%
1184 \def\glxstr@keylist{##1}%
1185 \@Glsxtr
1186 }

\@Glsxtr
1187 \newcommand*\@Glsxtr}[2] [] {%
1188 \ifglentryexists{##2}%
1189 {%
1190 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
1191 }%
1192 {%
1193 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
1194 description={\nopostdesc},##1}%
1195 }%
1196 \expandafter\gls\expandafter[\glxstr@keylist]{##2}%
1197 }

\glxstrpl
1198 \newcommand*\glxstrpl}[1] [] {%
1199 \def\glxstr@keylist{##1}%
1200 \@glxstrpl
1201 }

\@glxstrpl
1202 \newcommand*\@glxstrpl}[2] [] {%
1203 \ifglentryexists{##2}%
1204 {%
1205 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
1206 }%
1207 {%
1208 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
1209 description={\nopostdesc},##1}%
1210 }%
1211 \expandafter\glspl\expandafter[\glxstr@keylist]{##2}%

```



```
1212 }
```

```
\Glsxtrpl
```

```
1213 \newcommand*\Glsxtrpl[1][]{%
1214 \def\glxtr@keylist{##1}%
1215 \@Glsxtrpl
1216 }
```

```
\@Glsxtrpl
```

```
1217 \newcommand*\@Glsxtrpl[2][]{%
1218 \ifglstryexists{##2}
1219 {%
1220 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
1221 }%
1222 {%
1223 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
1224 description={\nopostdesc},##1}%
1225 }%
1226 \expandafter\Glspl\expandafter[\glxtr@keylist]{##2}%
1227 }
```

```
\GlsXtrWarning
```

```
1228 \newcommand*\GlsXtrWarning[2]{%
1229 \def\glxtr@optlist{##1}%
1230 \@onelevel@sanitize\glxtr@optlist
1231 \GlossariesExtraWarning{The options '\glxtr@optlist' have
1232 been ignored for entry '##2' as it has already been defined}%
1233 }
```

Disable commands after the glossary:

```
1234 \renewcommand\@printglossary[2]{%
1235 \def\glxtr@printglossopts{##1}%
1236 \@glxtr@orgprintglossary{##1}{##2}%
1237 \def\glxtr{\glxtr@disabledflycommand\glxtr}%
1238 \def\glxtrpl{\glxtr@disabledflycommand\glxtrpl}%
1239 \def\Glsxtr{\glxtr@disabledflycommand\Glsxtr}%
1240 \def\Glsxtrpl{\glxtr@disabledflycommand\Glsxtrpl}%
1241 }
```

```
abledflycommand
```

```
1242 \newcommand*\@glxtr@disabledflycommand[1]{%
1243 \PackageError{glossaries-extra}%
1244 {\string##1\space can't be used after any of the \MessageBreak
1245 glossaries have been displayed}%
1246 {The on-the-fly commands enabled by
1247 \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1248 before the glossaries. If you want to use any entries \MessageBreak
1249 after any of the glossaries, you must use the standard \MessageBreak
1250 method of first defining the entry and then using the \MessageBreak
```

```

1251      entry with commands like \string\gls}%
1252      \@@glxtr@disabledflycommand
1253  }%
1254  \newcommand*{\@@glxtr@disabledflycommand}[2][\{##2}

```

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

```

1255  \let\GlsXtrEnableOnTheFly\relax
1256 }
1257 \@onlypreamble\GlsXtrEnableOnTheFly

```

1.3.3 Existing Glossary Style Modifications

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

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

```

1258 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}

```

Modify \setglossarystyle to set the above.

`\setglossarystyle`

```

1259 \renewcommand*{\setglossarystyle}[1]{%
1260   \ifcsundef{@glsstyle@#1}%
1261   {%
1262     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
1263   }%
1264   {%
1265     \csname @glsstyle@#1\endcsname

```

Only set the current style if it exists.

```

1266   \protected@edef\@glxtr@current@style{#1}%
1267 }%
1268 \ifx\@glossary@default@style\relax
1269   \protected@edef\@glossary@default@style{#1}%
1270 \fi
1271 }

```

In case we have an old version of glossaries:

```

1272 \ifdef\@glossary@default@style
1273 {}
1274 {%
1275   \let\@glossary@default@style\relax
1276 }

```

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

```

1277 \ifdef\glslistdottedwidth
1278 {%
1279   \ifdim\glslistdottedwidth=.5\hsize

```

```

1280 \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1281 \AtBeginDocument{%
1282   \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1283     \setlength{\glslistdottedwidth}{.5\columnwidth}%
1284   \fi
1285 }%
1286 \fi
1287 }
1288 {}%

```

Similarly for \glsdescwidth:

\glsdescwidth

```

1289 \ifdef\glsdescwidth
1290 {%
1291   \ifdim\glsdescwidth=.6\hsize
1292     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1293   \AtBeginDocument{%
1294     \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1295       \setlength{\glsdescwidth}{.6\columnwidth}%
1296     \fi
1297   }%
1298 \fi
1299 }
1300 {}%

```

and for \glspagelistwidth:

\glspagelistwidth

```

1301 \ifdef\glspagelistwidth
1302 {%
1303   \ifdim\glspagelistwidth=.1\hsize
1304     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
1305   \AtBeginDocument{%
1306     \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1307       \setlength{\glspagelistwidth}{.1\columnwidth}%
1308     \fi
1309   }%
1310 \fi
1311 }
1312 {}%

```

aryentrynumbers Has the nonumberlist option been used?

```

1313 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1314 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1315   \glsnonumberlistfalse
1316   \renewcommand*{\glossaryentrynumbers}[1]{%
1317     \ifglstryexists{\glscurrententrylabel}%
1318     {%
1319       \@glsxtrpreloctag

```

```

1320     \GlsXtrFormatLocationList{#1}%
1321     \@glxtrpostloctag
1322     \gls@save@numberlist{#1}%
1323   }{}%
1324 }%
1325 \else
1326   \glsnonnumberlisttrue
1327   \renewcommand*{\glossaryentrynumbers}[1]{%
1328     \ifglseentryexists{\glscurrententrylabel}%
1329     {%
1330       \gls@save@numberlist{#1}%
1331     }{}%
1332   }%
1333 \fi

```

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

```

1334 \newcommand*{\GlsXtrFormatLocationList}[1]{#1}

```

Sometimes users want to prefix the location list with “page”/“pages”. The simplest way to determine if the location list consists of a single location is to check for instances of `\delimN` or `\delimR`, but this isn’t so easy to do as they might be embedded inside the argument of formatting commands. With a bit of trickery we can find out by adjusting `\delimN` and `\delimR` to set a flag and then save information to the auxiliary file for the next run.

`\ePreLocationTag`

```

1335 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
1336   \let\@glxtrpreloctag\@glxtrpreloctag
1337   \let\@glxtrpostloctag\@glxtrpostloctag
1338   \renewcommand*{\@glxtr@pagetag}{#1}%
1339   \renewcommand*{\@glxtr@pagetag}{#2}%
1340   \renewcommand*{\@glxtr@savepreloctag}[2]{%
1341     \csgdef{\@glxtr@preloctag##1}{##2}%
1342   }%
1343   \renewcommand*{\@glxtr@doloctag}{%
1344     \ifcsundef{\@glxtr@preloctag\glscurrententrylabel}%
1345     {%
1346       \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
1347         Rerun required}%
1348     }%
1349     {%
1350       \csuse{\@glxtr@preloctag\glscurrententrylabel}%
1351     }%
1352   }%
1353 }
1354 \@onlypreamble\GlsXtrEnablePreLocationTag

```

`\glxtrpreloctag`

```

1355 \newcommand*{\@@glsxtrpreloctag}{%
1356   \let\@glsxtr@org@delimN\delimN
1357   \let\@glsxtr@org@delimR\delimR
1358   \let\@glsxtr@org@glsignore\glsignore
      \gdef is required as the delimiters may occur inside a scope.
1359   \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1360   \renewcommand*{\delimN}{%
1361     \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1362     \@glsxtr@org@delimN}%
1363   \renewcommand*{\delimR}{%
1364     \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1365     \@glsxtr@org@delimR}%
1366   \renewcommand*{\glsignore}[1]{%
1367     \gdef\@glsxtr@thisloctag{\relax}%
1368     \@glsxtr@org@glsignore{##1}}%
1369   \@glsxtr@doloctag
1370 }

```

glsxtrpreloctag

```

1371 \newcommand*{\@glsxtrpreloctag}{%

```

@glsxtr@pagetag

```

1372 \newcommand*{\@glsxtr@pagetag}{%

```

glsxtr@pagetag

```

1373 \newcommand*{\@glsxtr@pagetag}{%

```

lsxtrpostloctag

```

1374 \newcommand*{\@@glsxtrpostloctag}{%
1375   \let\delimN\@glsxtr@org@delimN
1376   \let\delimR\@glsxtr@org@delimR
1377   \let\glsignore\@glsxtr@org@glsignore
1378   \protected@write\@auxout{%
1379     {\string\@glsxtr@savepreloctag{\glscurrententrylabel}\@glsxtr@thisloctag}}%
1380 }

```

lsxtrpostloctag

```

1381 \newcommand*{\@glsxtrpostloctag}{%

```

lsxtr@preloctag

```

1382 \newcommand*{\@glsxtr@savepreloctag}[2]{%
1383 \protected@write\@auxout{%
1384   \string\providecommand\string\@glsxtr@savepreloctag[2]{%

```

glsxtr@doloctag

```

1385 \newcommand*{\@glsxtr@doloctag}{%

```

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

```

1386 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
1387 \XKV@plfalse
1388 \XKV@sttrue
1389 \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
1390 {%
1391 \csname glsnonumberlist\XKV@resa\endcsname
1392 \ifglsnonumberlist
1393 \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1394 \else
1395 \def\glossaryentrynumbers##1{%
1396 \@glstrpreloctag
1397 \GlsXtrFormatLocationList{##1}%
1398 \@glstrpostloctag
1399 \gls@save@numberlist{##1}}%
1400 \fi
1401 }%
1402 }

```

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.

```

1403 \renewcommand*{\glsentryfmt}{%
1404 \ifglshasshort{\glslabel}{\glsssetabbrvfmt{\glscategory{\glslabel}}}%
1405 \glsifregular{\glslabel}%
1406 {\glstrregularfont{\glsgenentryfmt}}%
1407 {%
1408 \ifglshasshort{\glslabel}%
1409 {\glstrgenabbrvfmt}%
1410 {\glstrregularfont{\glsgenentryfmt}}%
1411 }%
1412 }

```

sxtrregularfont Font used for regular entries.

```

1413 \newcommand*{\glstrregularfont}[1]{#1}

```

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

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

```
1414 \renewcommand{\@gls@field@link}[4] [] {%
```

If the record option has been used, the information needs to be written to the aux file regardless of whether the enter exists.

```
1415 \@glstr@record{#2}{#3}{glslink}%
1416 \glsoifexists{#3}%
1417 {%
```

Save and restore the hyper setting (`\@gls@link` also does this, but that's too late if the optional argument of `\@gls@field@link` modifies it).

```
1418 \let\glstrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
1419 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1420 \def\glscustomtext{#4}%
1421 \@glstr@field@linkdefs
1422 #1%
1423 \@gls@link[#2]{#3}{#4}%
1424 \let\ifKV@glslink@hyper\glstrorg@ifKV@glslink@hyper
1425 }%
1426 \glspostlinkhook
1427 }
```

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

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

```
1428 \let\@glstr@org@gls@\@gls@
1429 \def\@gls@#1#2{%
1430 \@glstr@record{#1}{#2}{glslink}%
1431 \@glstr@org@gls@{#1}{#2}%
1432 }%
```

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

```
1433 \let\@glstr@org@glspl@\@glspl@
1434 \def\@glspl@#1#2{%
1435 \@glstr@record{#1}{#2}{glslink}%
1436 \@glstr@org@glspl@{#1}{#2}%
1437 }%
```

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

```
1438 \let\@glstr@org@Gls@\@Gls@
1439 \def\@Gls@#1#2{%
1440 \@glstr@record{#1}{#2}{glslink}%
1441 \@glstr@org@Gls@{#1}{#2}%
1442 }%
```

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

```
1443 \let\@glxtr@org@Glspl@\@Glspl@
1444 \def\@Glspl@#1#2{%
1445   \@glxtr@record{#1}{#2}{glslink}%
1446   \@glxtr@org@Glspl@{#1}{#2}%
1447 }%
```

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

```
1448 \let\@glxtr@org@GLS@\@GLS@
1449 \def\@GLS@#1#2{%
1450   \@glxtr@record{#1}{#2}{glslink}%
1451   \@glxtr@org@GLS@{#1}{#2}%
1452 }%
```

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

```
1453 \let\@glxtr@org@GLSpl@\@GLSpl@
1454 \def\@GLSpl@#1#2{%
1455   \@glxtr@record{#1}{#2}{glslink}%
1456   \@glxtr@org@GLSpl@{#1}{#2}%
1457 }%
```

`\@glsdisp` Save the original definition and redefine. Can't save and restore `\@glsdisp` since it has an optional argument.

```
1458 \renewcommand*{\@glsdisp}[3][{}]{%
1459   \@glxtr@record{#1}{#2}{glslink}%
1460   \glsdoifexists{#2}{%
1461     \let\do@gl@link@checkfirsthyper\@gl@link@checkfirsthyper
1462     \let\glsifplural\@secondoftwo
1463     \let\glscapscase\@firstofthree
1464     \def\glscustomtext{#3}%
1465     \def\glsinsert{}%
1466     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
1467     \@gl@link[#1]{#2}{\@glo@text}%
1468     \ifKV@glslink@local
1469       \glslocalunset{#2}%
1470     \else
1471       \glsunset{#2}%
1472     \fi
1473   }%
1474   \glspostlinkhook
1475 }
```

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

```
1476 \renewcommand*{\@gls@link}[3][{}]{%
1477   \@glxtr@record{#1}{#2}{glslink}%
1478   \glsdoifexistsordo{#2}%
1479   {%
1480     \let\do@gl@link@checkfirsthyper\relax
```



```

1481 \@gls@link[#1]{#2}{#3}%
1482 }%
1483 {%
1484 \glstextformat{#3}%
1485 }%
1486 \glspostlinkhook
1487 }

```

sxtrinitwrgloss Set the default if the wrgloss is omitted.

```

1488 \newcommand*{\glxsxtrinitwrgloss}{%
1489 \glsifattribute{\glslabel}{wrgloss}{after}%
1490 {%
1491 \glxsxtrinitwrglossbeforefalse
1492 }%
1493 {%
1494 \glxsxtrinitwrglossbeforetrue
1495 }%
1496 }

```

trwrglossbefore Conditional to determine if the indexing should be done before the link text.

```

1497 \newif\ifglxsxtrinitwrglossbefore
1498 \glxsxtrinitwrglossbeforetrue

```

Define a wrgloss key to determine whether to write the glossary information before or after the link text.

```

1499 \define@choicekey{glslink}{wrgloss}[\val\nr]{before,after}%
1500 {%
1501 \ifcase\nr\relax
1502 \glxsxtrinitwrglossbeforetrue
1503 \or
1504 \glxsxtrinitwrglossbeforefalse
1505 \fi
1506 }

```

\@gls@link Redefine to allow the indexing to be placed after the link text. By default this is done before the link text to prevent problems that can occur from the whatsit, but there may be times when the user would like the indexing done afterwards even though it causes a whatsit.

```

1507 \def\@gls@link[#1]#2#3{%
1508 \leavevmode
1509 \edef\glslabel{\glsdetoklabel{#2}}%
1510 \def\@gls@link@opts{#1}%
1511 \let\@gls@link@label\glslabel
1512 \def\@glsnumberformat{glsnumberformat}%
1513 \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
1514 \edef\glstype{\csname glo@\glslabel @type\endcsname}%
1515 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper

```

Initialise when indexing should occur (new to v1.14).

```

1516 \glxsxtrinitwrgloss

```

As the original definition. Note that the default link options may override \glxtrinitwrgloss.

```

1517 \@gls@setdefault@glslink@opts
1518 \do@gl:disablehyperinlist
1519 \do@gl@link@checkfirsthyper
1520 \setkeys{glslink}{#1}%
1521 \glslinkpostsetkeys
1522 \@gls@saveentrycounter
1523 \@gls@setsort{glslabel}%

```

Do write if it should occur before the link text:

```

1524 \ifglxtrinitwrglossbefore
1525   \do@wrglossary{#2}%
1526 \fi

```

Do the link text:

```

1527 \ifKV@glslink@hyper
1528   \@glslink{\glolinkprefix\glslabel}{\glstextformat{#3}}%
1529 \else
1530   \glndonohyperlink{\glolinkprefix\glslabel}{\glstextformat{#3}}%
1531 \fi

```

Do write if it should occur after the link text:

```

1532 \ifglxtrinitwrglossbefore
1533 \else
1534   \do@wrglossary{#2}%
1535 \fi

```

As the original definition:

```

1536 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
1537 }

```

```

1538 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}

```

```

1539 \define@key{glossadd}{theHvalue}{\def\@glxtr@theHvalue{#1}}

```

\glsadd Redefine to include \@glxtr@record

```

1540 \renewrobustcmd*{\glsadd}[2][\]{%
1541   \@gls@adjustmode
1542   \@glxtr@record{#1}{#2}{glossadd}%
1543   \glsoifexists{#2}%
1544   {%
1545     \def\@glsnumberformat{glsnumberformat}%
1546     \edef\@gls@counter{\csname glo@gl:detoklabel{#2}@counter\endcsname}%
1547     \def\@glxtr@thevalue{}%
1548     \def\@glxtr@theHvalue{\@glxtr@thevalue}%
1549     \setkeys{glossadd}{#1}%
1550     \ifdefempty{\@glxtr@thevalue}%
1551     {%
1552       \@gls@saveentrycounter
1553     }%

```

```

1554   {%
1555       \let\theglstrycounter\@glstr@thevalue
1556       \def\theHglstrycounter{\@glstr@theHvalue}%
1557   }%
1558   \@@do@wrglossary{#2}%
1559 }%
1560 }

```

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

```

1561 \newcommand*{\@glstr@field@linkdefs}{%
1562   \let\glstrifwasfirstuse\@secondoftwo
1563   \let\glstrifplural\@secondoftwo
1564   \let\glscapscase\@firstofthree
1565   \let\glstrinsert\@empty
1566 }

```

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

`assignfieldfont`

```

1567 \newcommand*{\glstrassignfieldfont}[1]{%
1568   \ifglstryexists{#1}%
1569   {%
1570     \ifglshasshort{#1}%
1571     {%
1572       \glsetabbrfmt{\glscategory{#1}}%
1573       \glstrifregular{#1}%
1574       {\let\@glstr@field@font\glstrregularfont}%
1575       {\let\@glstr@field@font\@firstofone}%
1576     }%
1577     {%
1578       \glstrifnotregular{#1}%
1579       {\let\@glstr@field@font\@firstofone}%
1580       {\let\@glstr@field@font\glstrregularfont}%
1581     }%
1582   }%
1583   {%
1584     \let\@glstr@field@font\@gobble
1585   }%
1586 }

```

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

```

1587 \def\@glstext@#1#2[#3]{%
1588   \glstrassignfieldfont{#2}%
1589   \@glstr@field@link{#1}{#2}{\@glstr@field@font{\glstr@field@font{#2}#3}}%
1590 }

```

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

```

1591 \def\@GLStext@#1#2[#3]{%

```

```

1592 \glstrassignfieldfont{#2}%
1593 \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
1594 {\@gls@field@font{\Glsaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
1595 }

```

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

```

1596 \def\@Glstext@#1#2[#3]{%
1597 \glstrassignfieldfont{#2}%
1598 \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
1599 {\@gls@field@font{\Glsaccesstext{#2}#3}}%
1600 }

```

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

`ecknohyperfirst`

```

1601 \newcommand*{\glstrchecknohyperfirst}[1]{%
1602 \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
1603 }

```

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

```

1604 \def\@glsfirst@#1#2[#3]{%
1605 \glstrassignfieldfont{#2}%
1606 \gls@field@link
1607 [\let\glstrifwasfirstuse\@firstoftwo
1608 \glstrchecknohyperfirst{#2}%
1609 ]{#1}{#2}%
1610 {\@gls@field@font{\glsaccessfirst{#2}#3}}%
1611 }

```

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

```

1612 \def\@Glsfirst@#1#2[#3]{%
1613 \glstrassignfieldfont{#2}%
1614 \gls@field@link
1615 [\let\glstrifwasfirstuse\@firstoftwo
1616 \let\glscapscase\@secondofthree
1617 \glstrchecknohyperfirst{#2}%
1618 ]%
1619 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
1620 }

```

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

```

1621 \def\@GLSfirst@#1#2[#3]{%
1622 \glstrassignfieldfont{#2}%

```

Ensure that \GLSfirst honours the nohyperfirst attribute.

```
1623 \@gls@field@link
1624 [\let\glstrifwasfirstuse\@firstoftwo
1625 \let\glscapscase\@thirdofthree
1626 \glstrchecknohyperfirst{#2}%
1627 ]%
1628 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1629 }
```

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

```
1630 \def\@glsplural@#1#2[#3]{%
1631 \glstrassignfieldfont{#2}%
1632 \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1633 {\@gls@field@font{\glsaccessplural{#2}#3}}%
1634 }
```

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

```
1635 \def\@Glsplural@#1#2[#3]{%
1636 \glstrassignfieldfont{#2}%
1637 \@gls@field@link
1638 [\let\glsifplural\@firstoftwo
1639 \let\glscapscase\@secondofthree
1640 ]%
1641 {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1642 }
```

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

```
1643 \def\@GLSplural@#1#2[#3]{%
1644 \glstrassignfieldfont{#2}%
1645 \@gls@field@link
1646 [\let\glsifplural\@firstoftwo
1647 \let\glscapscase\@thirdofthree
1648 ]%
1649 {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1650 }
```

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

```
1651 \def\glsfirstplural@#1#2[#3]{%
1652 \glstrassignfieldfont{#2}%
1653 \@gls@field@link
1654 [\let\glstrifwasfirstuse\@firstoftwo
1655 \let\glsifplural\@firstoftwo
1656 \glstrchecknohyperfirst{#2}%
1657 ]%
1658 {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1659 }
```

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

```
1660 \def\@Glsfirstplural@#1#2[#3]{%
1661   \glstrassignfieldfont{#2}%
    Ensure that \glfirstplural honours the nohyperfirst attribute.
1662   \@gls@field@link
1663   [\let\glstrifwasfirstuse\@firstoftwo
1664   \let\glstifplural\@firstoftwo
1665   \let\glscapscase\@secondofthree
1666   \glstrchecknohyperfirst{#2}%
1667   ]%
1668   {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1669 }
```

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

```
1670 \def\@GLSfirstplural@#1#2[#3]{%
1671   \glstrassignfieldfont{#2}%
    Ensure that \glfirstplural honours the nohyperfirst attribute.
1672   \@gls@field@link
1673   [\let\glstrifwasfirstuse\@firstoftwo
1674   \let\glstifplural\@firstoftwo
1675   \let\glscapscase\@thirdofthree
1676   \glstrchecknohyperfirst{#2}%
1677   ]%
1678   {#1}{#2}%
1679   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
1680 }
```

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

```
1681 \def\@glsname@#1#2[#3]{%
1682   \glstrassignfieldfont{#2}%
1683   \@gls@field@link{#1}{#2}{\@gls@field@font{\Glsaccessname{#2}#3}}%
1684 }
```

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

```
1685 \def\@Glsname@#1#2[#3]{%
1686   \glstrassignfieldfont{#2}%
1687   \@gls@field@link
1688   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1689   {\@gls@field@font{\Glsaccessname{#2}#3}}%
1690 }
```

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

```
1691 \def\@GLSname@#1#2[#3]{%
1692   \glstrassignfieldfont{#2}%
1693   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1694   {#1}{#2}%
1695   {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
1696 }
```

```

\@Glsdesc@
1697 \def\@Glsdesc@#1#2[#3]{%
1698   \glstrassignfieldfont{#2}%
1699   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
1700 }

\@GLSdesc@   First letter uppercase version.
1701 \def\@GLSdesc@#1#2[#3]{%
1702   \glstrassignfieldfont{#2}%
1703   \@gls@field@link
1704   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1705   {\@gls@field@font{\GLSaccessdesc{#2}#3}}%
1706 }

\@GLSdesc@   All uppercase version.
1707 \def\@GLSdesc@#1#2[#3]{%
1708   \glstrassignfieldfont{#2}%
1709   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1710   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
1711 }

@Glsdescplural@   No case-changing version.
1712 \def\@Glsdescplural@#1#2[#3]{%
1713   \glstrassignfieldfont{#2}%
1714   \@gls@field@link
1715   [\let\glscapscase\@secondoftwo
1716   \let\glsifplural\@firstoftwo
1717   ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
1718 }

@Glsdescplural@   First letter uppercase version.
1719 \def\@Glsdescplural@#1#2[#3]{%
1720   \glstrassignfieldfont{#2}%
1721   \@gls@field@link
1722   [\let\glscapscase\@secondoftwo
1723   \let\glsifplural\@firstoftwo
1724   ]{#1}{#2}{\@gls@field@font{\GLSaccessdescplural{#2}#3}}%
1725 }

@GLSdescplural@   All uppercase version.
1726 \def\@GLSdesc@#1#2[#3]{%
1727   \glstrassignfieldfont{#2}%
1728   \@gls@field@link
1729   [\let\glscapscase\@thirdoftwo
1730   \let\glsifplural\@firstoftwo
1731   ]%
1732   {#1}{#2}%
1733   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
1734 }

```

\@glssymbol@

```
1735 \def\@glssymbol@#1#2[#3]{%
1736   \glstrassignfieldfont{#2}%
1737   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
1738 }
```

\@Glsymbol@ First letter uppercase version.

```
1739 \def\@Glsymbol@#1#2[#3]{%
1740   \glstrassignfieldfont{#2}%
1741   \@gls@field@link
1742   [\let\glscapscase\@secondoftwo]%
1743   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
1744 }
```

\@GLSsymbol@ All uppercase version.

```
1745 \def\@GLSsymbol@#1#2[#3]{%
1746   \glstrassignfieldfont{#2}%
1747   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1748   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
1749 }
```

lssymbolplural@ No case-changing version.

```
1750 \def\@lssymbolplural@#1#2[#3]{%
1751   \glstrassignfieldfont{#2}%
1752   \@gls@field@link
1753   [\let\glscapscase\@secondoftwo
1754   \let\glsifplural\@firstoftwo
1755   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
1756 }
```

lssymbolplural@ First letter uppercase version.

```
1757 \def\@lssymbolplural@#1#2[#3]{%
1758   \glstrassignfieldfont{#2}%
1759   \@gls@field@link
1760   [\let\glscapscase\@secondoftwo
1761   \let\glsifplural\@firstoftwo
1762   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
1763 }
```

LSsymbolplural@ All uppercase version.

```
1764 \def\@LSsymbol@#1#2[#3]{%
1765   \glstrassignfieldfont{#2}%
1766   \@gls@field@link
1767   [\let\glscapscase\@thirdoftwo
1768   \let\glsifplural\@firstoftwo
1769   ]%
1770   {#1}{#2}%
1771   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
1772 }
```


\@Glsuseri@ First letter uppercase version.

```
1773 \def\@Glsuseri@#1#2[#3]{%
1774   \glstrassignfieldfont{#2}%
1775   \@gls@field@link
1776   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1777   {\@gls@field@font{\Glsentryuseri{#2}#3}}%
1778 }
```

\@GLSuseri@ All uppercase version.

```
1779 \def\@GLSuseri@#1#2[#3]{%
1780   \glstrassignfieldfont{#2}%
1781   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1782   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseri{#2}#3}}}%
1783 }
```

\@Glsuserii@ First letter uppercase version.

```
1784 \def\@Glsuserii@#1#2[#3]{%
1785   \glstrassignfieldfont{#2}%
1786   \@gls@field@link
1787   [\let\glscapscase\@secondoftwo]%
1788   {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
1789 }
```

\@GLSuserii@ All uppercase version.

```
1790 \def\@GLSuserii@#1#2[#3]{%
1791   \glstrassignfieldfont{#2}%
1792   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1793   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
1794 }
```

\@Glsuseriii@ First letter uppercase version.

```
1795 \def\@Glsuseriii@#1#2[#3]{%
1796   \glstrassignfieldfont{#2}%
1797   \@gls@field@link
1798   [\let\glscapscase\@secondoftwo]%
1799   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
1800 }
```

\@GLSuseriii@ All uppercase version.

```
1801 \def\@GLSuseriii@#1#2[#3]{%
1802   \glstrassignfieldfont{#2}%
1803   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1804   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
1805 }
```

\@Glsuseriv@ First letter uppercase version.

```
1806 \def\@Glsuseriv@#1#2[#3]{%
1807   \glstrassignfieldfont{#2}%
1808 }
```

```

1808 \@gls@field@link
1809 [\let\glscapscase\@secondoftwo]%
1810 {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
1811 }

```

\@GLSuseriv@ All uppercase version.

```

1812 \def\@GLSuseriv@#1#2[#3]{%
1813 \glstrassignfieldfont{#2}%
1814 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1815 {#1}{#2}%
1816 {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
1817 }

```

\@Glsuserv@ First letter uppercase version.

```

1818 \def\@Glsuserv@#1#2[#3]{%
1819 \glstrassignfieldfont{#2}%
1820 \@gls@field@link
1821 [\let\glscapscase\@secondoftwo]%
1822 {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
1823 }

```

\@GLSuserv@ All uppercase version.

```

1824 \def\@GLSuserv@#1#2[#3]{%
1825 \glstrassignfieldfont{#2}%
1826 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1827 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
1828 }

```

\@Glsuservi@ First letter uppercase version.

```

1829 \def\@Glsuservi@#1#2[#3]{%
1830 \glstrassignfieldfont{#2}%
1831 \@gls@field@link
1832 [\let\glscapscase\@secondoftwo]%
1833 {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}%
1834 }

```

\@GLSuservi@ All uppercase version.

```

1835 \def\@GLSuservi@#1#2[#3]{%
1836 \glstrassignfieldfont{#2}%
1837 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1838 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
1839 }

```

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

\@acrshort No case change.

```

1840 \def\@acrshort#1#2[#3]{%

```

```

1841 \glsdoifexists{#2}%
1842 {%
1843   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1844   \let\glxtrifwasfirstuse\@secondoftwo
1845   \let\gl@ifplural\@secondoftwo
1846   \let\glscapscase\@firstofthree
1847   \let\glinsert\@empty
1848   \def\glscustomtext{%
1849     \acronymfont{\glaccessshort{#2}}#3%
1850   }%
1851   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1852 }%
1853 \glspostlinkhook
1854 }

```

\@Acrshort First letter uppercase.

```

1855 \def\@Acrshort#1#2[#3]{%
1856   \glsdoifexists{#2}%
1857   {%
1858     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1859     \let\glxtrifwasfirstuse\@secondoftwo
1860     \let\gl@ifplural\@secondoftwo
1861     \let\glscapscase\@secondofthree
1862     \let\glinsert\@empty
1863     \def\glscustomtext{%
1864       \acronymfont{\Glsaccessshort{#2}}#3%
1865     }%
1866     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1867   }%
1868   \glspostlinkhook
1869 }

```

\@ACRshort All uppercase.

```

1870 \def\@ACRshort#1#2[#3]{%
1871   \glsdoifexists{#2}%
1872   {%
1873     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1874     \let\glxtrifwasfirstuse\@secondoftwo
1875     \let\gl@ifplural\@secondoftwo
1876     \let\glscapscase\@thirdofthree
1877     \let\glinsert\@empty
1878     \def\glscustomtext{%
1879       \mfirstucMakeUppercase{\acronymfont{\glaccessshort{#2}}#3}%
1880     }%
1881     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1882   }%
1883   \glspostlinkhook
1884 }

```

\@acrshortpl No case change.

```
1885 \def\@acrshortpl#1#2[#3]{%
1886   \glsdoifexists{#2}%
1887   {%
1888     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1889     \let\glxtrifwasfirstuse\@secondoftwo
1890     \let\gl@sifplural\@firstoftwo
1891     \let\glscapscase\@firstofthree
1892     \let\gl$insert\@empty
1893     \def\glscustomtext{%
1894       \acronymfont{\gl@saccesssshortpl{#2}}#3%
1895     }%
1896     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1897   }%
1898   \glspostlinkhook
1899 }
```

\@Acrshortpl First letter uppercase.

```
1900 \def\@Acrshortpl#1#2[#3]{%
1901   \glsdoifexists{#2}%
1902   {%
1903     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1904     \let\glxtrifwasfirstuse\@secondoftwo
1905     \let\gl@sifplural\@firstoftwo
1906     \let\glscapscase\@secondofthree
1907     \let\gl$insert\@empty
1908     \def\glscustomtext{%
1909       \acronymfont{\Glsaccesssshortpl{#2}}#3%
1910     }%
1911     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1912   }%
1913   \glspostlinkhook
1914 }
```

\@ACRshortpl All uppercase.

```
1915 \def\@ACRshortpl#1#2[#3]{%
1916   \glsdoifexists{#2}%
1917   {%
1918     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1919     \let\glxtrifwasfirstuse\@secondoftwo
1920     \let\gl@sifplural\@firstoftwo
1921     \let\glscapscase\@thirdofthree
1922     \let\gl$insert\@empty
1923     \def\glscustomtext{%
1924       \mfirstucMakeUppercase{\acronymfont{\gl@saccesssshortpl{#2}}#3}%
1925     }%
1926     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1927   }%
1928   \glspostlinkhook
```

1929 }

\@acrlong No case change.

```
1930 \def\@acrlong#1#2[#3]{%
1931   \glsdoifexists{#2}%
1932   {%
1933     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1934     \let\glxtrifwasfirstuse\@secondoftwo
1935     \let\glsifplural\@secondoftwo
1936     \let\glscapscase\@firstofthree
1937     \let\glsinsert\@empty
1938     \def\glscustomtext{%
1939       \acronymfont{\glsaccesslong{#2}}#3%
1940     }%
1941     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1942   }%
1943   \glspostlinkhook
1944 }
```

\@Acrlong First letter uppercase.

```
1945 \def\@Acrlong#1#2[#3]{%
1946   \glsdoifexists{#2}%
1947   {%
1948     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1949     \let\glxtrifwasfirstuse\@secondoftwo
1950     \let\glsifplural\@secondoftwo
1951     \let\glscapscase\@secondofthree
1952     \let\glsinsert\@empty
1953     \def\glscustomtext{%
1954       \acronymfont{\Glsaccesslong{#2}}#3%
1955     }%
1956     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1957   }%
1958   \glspostlinkhook
1959 }
```

\@ACRlong All uppercase.

```
1960 \def\@ACRlong#1#2[#3]{%
1961   \glsdoifexists{#2}%
1962   {%
1963     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1964     \let\glxtrifwasfirstuse\@secondoftwo
1965     \let\glsifplural\@secondoftwo
1966     \let\glscapscase\@thirdofthree
1967     \let\glsinsert\@empty
1968     \def\glscustomtext{%
1969       \mfirstucMakeUppercase{\acronymfont{\glsaccesslong{#2}}#3}%
1970     }%
1971     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1972   }
```

```

1972 }%
1973 \glspostlinkhook
1974 }

```

\@acrlongpl No case change.

```

1975 \def\@acrlongpl#1#2[#3]{%
1976   \glsdoidexists{#2}%
1977   {%
1978     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1979     \let\glxtrifwasfirstuse\@secondoftwo
1980     \let\gl@sifplural\@firstoftwo
1981     \let\glscapscase\@firstofthree
1982     \let\gl$insert\@empty
1983     \def\glscustomtext{%
1984       \acronymfont{\gl@saccesslongpl{#2}}#3%
1985     }%
1986     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1987   }%
1988   \glspostlinkhook
1989 }

```

\@Acrlongpl First letter uppercase.

```

1990 \def\@Acrlongpl#1#2[#3]{%
1991   \glsdoidexists{#2}%
1992   {%
1993     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1994     \let\glxtrifwasfirstuse\@secondoftwo
1995     \let\gl@sifplural\@firstoftwo
1996     \let\glscapscase\@secondofthree
1997     \let\gl$insert\@empty
1998     \def\glscustomtext{%
1999       \acronymfont{\Glsaccesslongpl{#2}}#3%
2000     }%
2001     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2002   }%
2003   \glspostlinkhook
2004 }

```

\@ACRlongpl All uppercase.

```

2005 \def\@ACRlongpl#1#2[#3]{%
2006   \glsdoidexists{#2}%
2007   {%
2008     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2009     \let\glxtrifwasfirstuse\@secondoftwo
2010     \let\gl@sifplural\@firstoftwo
2011     \let\glscapscase\@thirdofthree
2012     \let\gl$insert\@empty
2013     \def\glscustomtext{%
2014       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslongpl{#2}}#3}%

```

```

2015 }%
2016 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2017 }%
2018 \glspostlinkhook
2019 }

```

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

\@glsaddkey

```

2020 \renewcommand*{\@glsaddkey}[7]{%
2021   \key@ifundefined{glossentry}{#1}%
2022   {%
2023     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
2024     \appto\@gls@keymap{,{#1}{#1}}%
2025     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
2026     \appto\@newglossaryentryposthook{%
2027       \letcs{\@glo@tmp}{@glo@#1}%
2028       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
2029     }%
2030     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
2031     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

2032   \ifcsdef{@gls@user@#1@}%
2033   {%
2034     \PackageError{glossaries}%
2035     {Can't define '\string#5' as helper command
2036     '\expandafter\string\csname @gls@user@#1@\endcsname' already
2037     exists}%
2038   }%
2039 }%
2040 {%
2041   \expandafter\newcommand\expandafter*\expandafter
2042   {\csname @gls@user@#1\endcsname}[2][ ]{%
2043     \new@ifnextchar[%
2044       {\csuse{@gls@user@#1@}{##1}{##2}}%
2045       {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
2046   \csdef{@gls@user@#1@}##1##2[##3]{%
2047     \@gls@field@link{##1}{##2}{#3{##2}##3}%
2048   }%
2049   \newrobustcmd*{#5}{%
2050     \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
2051   }%

```

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

```

2052   \ifcsdef{@Gls@user@#1@}%
2053   {%
2054     \PackageError{glossaries}%
2055     {Can't define '\string#6' as helper command

```

```

2056         '\expandafter\string\csname @Gls@user@#1@\endcsname' already
2057         exists}%
2058     {}%
2059 }%
2060 {%
2061     \expandafter\newcommand\expandafter*\expandafter
2062     {\csname @Gls@user@#1@\endcsname}[2][\%
2063         \new@ifnextchar[%
2064             {\csuse{@Gls@user@#1@}{##1}{##2}}}%
2065             {\csuse{@Gls@user@#1@}{##1}{##2}[]}}}%
2066     \csdef{@Gls@user@#1@}##1##2[##3]{%
2067         \@gls@field@link[\let\gls@caps@case\@secondofthree]%
2068         {##1}{##2}{#4{##2}##3}%
2069     }%
2070     \newrobustcmd*{#6}{%
2071         \expandafter\@gls@hyp@opt\csname @Gls@user@#1@\endcsname}%
2072     }%

```

Finally the all caps version (modified):

```

2073     \ifcsdef{@GLS@user@#1@}%
2074     {%
2075         \PackageError{glossaries}%
2076         {Can't define '\string#7' as helper command
2077         '\expandafter\string\csname @GLS@user@#1@\endcsname' already
2078         exists}%
2079     {}%
2080 }%
2081 {%
2082     \expandafter\newcommand\expandafter*\expandafter
2083     {\csname @GLS@user@#1@\endcsname}[2][\%
2084         \new@ifnextchar[%
2085             {\csuse{@GLS@user@#1@}{##1}{##2}}}%
2086             {\csuse{@GLS@user@#1@}{##1}{##2}[]}}}%
2087     \csdef{@GLS@user@#1@}##1##2[##3]{%
2088         \@gls@field@link[\let\gls@caps@case\@thirdofthree]%
2089         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}}%
2090     }%
2091     \newrobustcmd*{#7}{%
2092         \expandafter\@gls@hyp@opt\csname @GLS@user@#1@\endcsname}%
2093     }%
2094 }%
2095 {%
2096     \PackageError{glossaries-extra}{Key '#1' already exists}{}%
2097 }%
2098 }

```

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

```

2099 \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.

```
2100 \let\@glstr@org@checkfirsthyper\@gls@link@checkfirsthyper
2101 \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.

```
2102 \ifglsused{\glslabel}%
2103 {\let\glstr@ifwasfirstuse\@secondoftwo}
2104 {\let\glstr@ifwasfirstuse\@firstoftwo}%
```

Store the category label for convenience.

```
2105 \edef\glscategorylabel{\glscategory{\glslabel}}%
2106 \ifglsused{\glslabel}%
2107 {%
2108   \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
2109   {\KV@glslink@hyperfalse}{}%
2110 }%
2111 {%
2112   \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
2113   {\KV@glslink@hyperfalse}{}%
2114 }%
2115 \glslinkcheckfirsthyperhook
2116 }
```

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.

```
2117 \ifdef\do@glstdisablehyperinlist
2118 {%
2119   \let\@glstr@do@glstdisablehyperinlist\do@glstdisablehyperinlist
2120   \renewcommand*{\do@glstdisablehyperinlist}{%
2121     \@glstr@do@glstdisablehyperinlist
2122     \glsifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}%
2123   }
2124 }
2125 }
```

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

```
2126 \define@boolkey{glslink}{noindex}[true]{}
2127 \KV@glslink@noindexfalse
```

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

lt@glslink@opts

```
2128 \ifdef\@gls@setdefault@glslink@opts
2129 {
2130   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2131     \KV@glslink@noindexfalse
```

```

2132 \glstrsetaliasnoindex
2133 }
2134 }
2135 {

```

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

```

2136 \newcommand*{\@gl@setdefault@gl@link@opts}{%
2137 \KV@gl@link@noindexfalse
2138 \glstrsetaliasnoindex
2139 }
2140 \pretoto\do@gl:disablehyperinlist{\@gl@setdefault@gl@link@opts}
2141 }

```

setaliasnoindex Allow user to hook into the alias noindex setting. Default behaviour switches off indexing for aliases. If the record option is on, this will have been defined to do nothing. (bib2gls will deal with records for aliased entries.)

```

2142 \providecommand*\glstrsetaliasnoindex{%
2143 \KV@gl@link@noindextrue
2144 }

```

setaliasnoindex

```

2145 \newcommand*{\@glstrsetaliasnoindex}{%
2146 \ifglshasfield{alias}{\glslabel}%
2147 {%
2148 \let\glstrindexaliased\@glstrindexaliased
2149 \glstrsetaliasnoindex
2150 \let\glstrindexaliased\@no@glstrindexaliased
2151 }%
2152 }%
2153 }

```

xtrindexaliased

```

2154 \newcommand{\@glstrindexaliased}{%
2155 \ifKV@gl@link@noindex
2156 \else
2157 \begin{group}
2158 \def\@gl@numberformat{gl@numberformat}%
2159 \edef\@gl@counter{\csname gl@gl@detoklabel{\glslabel}@counter\endcsname}%
2160 \glstr@saveentrycounter
2161 \do@wrglossary{\glstralias{\glslabel}}%
2162 \endgroup
2163 \fi
2164 }

```

xtrindexaliased

```

2165 \newcommand{\@no@glstrindexaliased}{%
2166 \PackageError{glossaries-extra}{\string\glstrindexaliased\space
2167 not permitted outside definition of \string\glstrsetaliasnoindex}%
2168 {}%
2169 }

```

`\glstrindexaliased` Provide a command to redirect alias indexing, but only allow it to be used within `\glstrsetaliasnoindex`.

```
2170 \let\glstrindexaliased\@no\glstrindexaliased
```

`\setdefaultGlsOpts` Set the default options for `\glslink` etc.

```
2171 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
2172   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2173     \setkeys{glslink}{#1}%
2174     \@glstrsetaliasnoindex
2175   }%
2176 }
```

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

```
2177 \newcommand*{\glstrifindexing}[2]{%
2178   \ifKV@glslink@noindex #2\else #1\fi
2179 }
```

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

```
2180 \renewcommand*{\glswriteentry}[2]{%
2181   \glstrifindexing
2182   {%
2183     \ifglindexonlyfirst
2184       \ifglused{#1}
2185       {\glstrdoautoindexname{#1}{dualindex}}%
2186       {#2}%
2187     \else
2188       \gl@ifattribute{#1}{indexonlyfirst}{true}%
2189       {\ifglused{#1}
2190        {\glstrdoautoindexname{#1}{dualindex}}%
2191        {#2}}%
2192       {#2}%
2193     \fi
2194   }%
2195   {}%
2196 }
```

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

```
2197 \appto\do@wrglossary{\@glstrdo@wrindex
2198   \glstrdowrglossaryhook{\@gls@label}%
2199 }
```

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

Similarly for the “noidx” version:

`\s@noidxglossary`

```
2200 \appto\gls@noidxglossary{\@glstrdo@wrindex
2201   \glstrdowrglossaryhook{\@gls@label}%
2202 }
```

xtr@do@@wrindex

```
2203 \newcommand*{\@glsxtr@do@@wrindex}{%
2204   \glsxtrdoautoindexname{\@gls@label}{dualindex}%
2205 }
```

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.)

```
2206 \newcommand*{\glsxtrdowrglossaryhook}[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.

```
2207 \newcommand*{\@gls@alt@hyp@opt}[1]{%
2208   \let\glslinkvar\@firstofthree
2209   \let\@gls@hyp@opt@cs#1\relax
2210   \@ifstar{\s@gls@hyp@opt}%
2211   {\@ifnextchar+%
2212     {\@firstoftwo{\p@gls@hyp@opt}}%
2213     {%
2214       \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
2215       {\@firstoftwo{\@alt@gls@hyp@opt}}%
2216       {#1}%
2217     }%
2218   }%
2219 }
```

alt@gls@hyp@opt User version

```
2220 \newcommand*{\@alt@gls@hyp@opt}[1] [] {%
2221   \let\glslinkvar\@firstofthree
2222   \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}
```

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

```
2223 \newcommand*{\@gls@alt@hyp@opt@char}{}
```

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

```
2224 \newcommand*{\@gls@alt@hyp@opt@keys}{}
```

rSetAltModifier

```
2225 \newcommand*{\GlsXtrSetAltModifier}[2]{%
2226   \let\@gls@hyp@opt\@gls@alt@hyp@opt
2227   \def\@gls@alt@hyp@opt@char{#1}%
2228   \def\@gls@alt@hyp@opt@keys{#2}%
2229 }
```

\glsdohyperlink Unpleasant complications can occur if the text or first key etc contains \gls, particularly if there are hyperlinks. To get around this problem, patch \glsdohyperlink so that it temporarily makes \gls behave like \glstext[*hyper=false,noindex*]. (This will be overridden if the user explicitly cancels either of those options in the optional argument of \gls

or using the plus version.) This also patches the short form commands like `\acrshort` and `\glxtrshort` to use `\glentryshort` and, similarly, the long form commands like `\acrlong` and `\glxtrlong` to use `\glentrylong`. Added attribute check.

```

2230 \renewcommand*{\glsdohyperlink}[2]{%
2231   \glshasattribute{\glslabel}{targeturl}%
2232   {%
2233     \glshasattribute{\glslabel}{targetname}%
2234     {%
2235       \glshasattribute{\glslabel}{targetcategory}%
2236       {%
2237         \hyperref{\glsggetattribute{\glslabel}{targeturl}}{%
2238           {\glsggetattribute{\glslabel}{targetcategory}}}%
2239         {\glsggetattribute{\glslabel}{targetname}}}%
2240         {{\glxtrprotectlinks#2}}}%
2241       }%
2242     }%
2243     \hyperref{\glsggetattribute{\glslabel}{targeturl}}{%
2244       }%
2245       {\glsggetattribute{\glslabel}{targetname}}}%
2246       {{\glxtrprotectlinks#2}}}%
2247   }%
2248 }%
2249 {%
2250   \href{\glsggetattribute{\glslabel}{targeturl}}{%
2251     {{\glxtrprotectlinks#2}}}%
2252   }%
2253 }%
2254 {%

```

Check for alias.

```

2255   \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
2256   \ifdefvoid\gloaliaslabel
2257   {%
2258     \hyperlink{#1}{{\glxtrprotectlinks#2}}}%
2259   }%
2260   {%

```

Redirect link to the alias target.

```

2261     \hyperlink
2262     {\glolinkprefix\glsetoklabel{\gloaliaslabel}}%
2263     {{\glxtrprotectlinks#2}}}%
2264   }%
2265 }%
2266 }

```

`glsdisablehyper` Redefine to set `\glslabel` (to allow it to be picked up by `\glsdohyperlink`). Also made it robust and added grouping to localise the definition of `\glslabel`. The original internal command `@glo@label` could probably be simply replaced with `\glslabel`, but it's retained in case its removal causes unexpected problems.

```

2267 \renewrobustcmd*{\gls hyperlink}[2][\glsentrytext{\@glo@label}]{%
2268 \def\@glo@label{#2}%
2269 {\edef\glslabel{#2}%
2270 \@glslink{\glo@linkprefix\glslabel}{#1}}%
2271 }

```

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

```

2272 \ifundef\glsdonohyperlink
2273 {%
2274 \renewcommand{\glsdisablehyper}{%
2275 \KV@glslink@hyperfalse
2276 \let\@glslink\glsdonohyperlink
2277 \let\@glstarget\@secondoftwo
2278 }
2279 }
2280 {}

```

`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. The generated text is scoped.

```

2281 \def\glsdonohyperlink#1#2{{\glsxtrprotectlinks #2}}

```

Reset `\@glslink` with patched versions:

```

2282 \ifcsundef{hyperlink}%
2283 {%
2284 \let\@glslink\glsdonohyperlink
2285 }%
2286 {%
2287 \let\@glslink\glsdohyperlink
2288 }

```

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

```

2289 \newcommand*{\glsxtrprotectlinks}{%
2290 \KV@glslink@hyperfalse
2291 \KV@glslink@noindextrue
2292 \let\@gls@\@glsxtr@p@text@
2293 \let\@Gls@\@Glsxtr@p@text@
2294 \let\@GLS@\@GLSxtr@p@text@
2295 \let\@glspl@\@glsxtr@p@plural@
2296 \let\@Glspl@\@Glsxtr@p@plural@
2297 \let\@GLSpl@\@GLSxtr@p@plural@
2298 \let\@glsxtrshort@\@glsxtr@p@short@
2299 \let\@Glsxtrshort@\@Glsxtr@p@short@
2300 \let\@GLSxtrshort@\@GLSxtr@p@short@
2301 \let\@glsxtrlong@\@glsxtr@p@long@
2302 \let\@Glsxtrlong@\@Glsxtr@p@long@
2303 \let\@GLSxtrlong@\@GLSxtr@p@long@

```

```

2304 \let\@glxstrshortpl\@glxstrp@shortpl@
2305 \let\@Glsxtrshortpl\@Glsxtrp@shortpl@
2306 \let\@GLSxtrshortpl\@GLSxtrp@shortpl@
2307 \let\@glxstrlongpl\@glxstrp@longpl@
2308 \let\@Glsxtrlongpl\@Glsxtrp@longpl@
2309 \let\@GLSxtrlongpl\@GLSxtrp@longpl@
2310 \let\@acrshort\@glxstrp@acrshort@
2311 \let\@Acrshort\@Glsxtrp@acrshort@
2312 \let\@ACRshort\@GLSxtrp@acrshort@
2313 \let\@acrshortpl\@glxstrp@acrshortpl@
2314 \let\@Acrshortpl\@Glsxtrp@acrshortpl@
2315 \let\@ACRshortpl\@GLSxtrp@acrshortpl@
2316 \let\@acrlong\@glxstrp@acrlong@
2317 \let\@Acrlong\@Glsxtrp@acrlong@
2318 \let\@ACRlong\@GLSxtrp@acrlong@
2319 \let\@acrlongpl\@glxstrp@acrlongpl@
2320 \let\@Acrlongpl\@Glsxtrp@acrlongpl@
2321 \let\@ACRlongpl\@GLSxtrp@acrlongpl@
2322 }

```

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

@glxstrp@text@

```
2323 \def\@glxstrp@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}
```

@Glsxtrp@text@

```
2324 \def\@Glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtrp@text@

```
2325 \def\@GLSxtrp@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2326 \def\@lsxtrp@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2327 \def\@lsxtrp@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

LSxtrp@plural@

```
2328 \def\@LSxtrp@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glxstrp@short@

```

2329 \def\@glxstrp@short@#1#2[#3]{%
2330 {%
2331   \glssetabbrvfmt{\glscategory{#2}}%
2332   \glsabbrvfont{\glsentryshort{#2}}#3%
2333 }%
2334 }

```

Glsxtr@p@short@

```
2335 \def\@Glsxtr@p@short@#1#2[#3]{%
2336 {%
2337   \glsetabbrvfmt{\glscategory{#2}}%
2338   \glsabbrvfont{\Glsentryshort{#2}}#3%
2339 }%
2340 }
```

GLSxtr@p@short@

```
2341 \def\@GLSxtr@p@short@#1#2[#3]{%
2342 {%
2343   \glsetabbrvfmt{\glscategory{#2}}%
2344   \mfirstucMakeUppercase{\glsabbrvfont{\Glsentryshort{#2}}#3}%
2345 }%
2346 }
```

sxtr@p@shortpl@

```
2347 \def\@glxtr@p@shortpl@#1#2[#3]{%
2348 {%
2349   \glsetabbrvfmt{\glscategory{#2}}%
2350   \glsabbrvfont{\Glsentryshortpl{#2}}#3%
2351 }%
2352 }
```

Sxtr@p@shortpl@

```
2353 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2354 {%
2355   \glsetabbrvfmt{\glscategory{#2}}%
2356   \glsabbrvfont{\Glsentryshortpl{#2}}#3%
2357 }%
2358 }
```

Sxtr@p@shortpl@

```
2359 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2360 {%
2361   \glsetabbrvfmt{\glscategory{#2}}%
2362   \mfirstucMakeUppercase{\glsabbrvfont{\Glsentryshortpl{#2}}#3}%
2363 }%
2364 }
```

@glxtr@p@long@

```
2365 \def\@glxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@Glsxtr@p@long@

```
2366 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@GLSxtr@p@long@

```
2367 \def\@GLSxtr@p@long@#1#2[#3]{%
2368   {\mfirstucMakeUppercase{\glslongfont{\Glsentrylong{#2}}#3}}}
```


lsxtr@p@longpl@
2369 \def\@glxtr@p@longpl@#1#2[#3]{\glentrylongpl{#2}#3}}

lsxtr@p@longpl@
2370 \def\@Glsxtr@p@longpl@#1#2[#3]{\glslongfont{\Glentrylongpl{#2}#3}}

LSxtr@p@longpl@
2371 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2372 {\mfirstucMakeUppercase{\glslongfont{\glentrylongpl{#2}#3}}}

xtr@p@acrshort@
2373 \def\@glxtr@p@acrshort@#1#2[#3]{\acronymfont{\glentryshort{#2}#3}}

xtr@p@acrshort@
2374 \def\@Glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glentryshort{#2}#3}}

xtr@p@acrshort@
2375 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2376 {\mfirstucMakeUppercase{\acronymfont{\glentryshort{#2}#3}}}

r@p@acrshortpl@
2377 \def\@glxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\glentryshortpl{#2}#3}}

r@p@acrshortpl@
2378 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glentryshortpl{#2}#3}}

r@p@acrshortpl@
2379 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2380 {\mfirstucMakeUppercase{\acronymfont{\glentryshortpl{#2}#3}}}

sxtr@p@acrlong@
2381 \def\@glxtr@p@acrlong@#1#2[#3]{\glentrylong{#2}#3}}

sxtr@p@acrlong@
2382 \def\@Glsxtr@p@acrlong@#1#2[#3]{\Glentrylong{#2}#3}}

Sxtr@p@acrlong@
2383 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2384 {\mfirstucMakeUppercase{\glentrylong{#2}#3}}}

tr@p@acrlongpl@
2385 \def\@glxtr@p@acrlongpl@#1#2[#3]{\glentrylongpl{#2}#3}}

tr@p@acrlongpl@
2386 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\Glentrylongpl{#2}#3}}

tr@p@acrlongpl@

```
2387 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2388 {\mfirstucMakeUppercase{\glstrylongpl{#2}#3}}}
```

Commands to minimise conflict.

\@glxtrp@opt

```
2389 \newcommand*{\@glxtrp@opt}{hyper=false,noindex}
```

\glxtrsetopts Used in glossary to switch hyperlinks on for the \@glxtrp type of commands.

```
2390 \newcommand*{\glxtrsetopts}[1]{%
2391 \renewcommand*{\@glxtrp@opt}{#1}%
2392 }
```

lossxtrsetopts Used in glossary to switch hyperlinks on for the \@glxtrp type of commands.

```
2393 \newcommand*{\glossxtrsetopts}{%
2394 \glxtrsetopts{noindex}%
2395 }
```

\@@glxtrp

```
2396 \newrobustcmd*{\@@glxtrp}[2]{%
```

Add scope.

```
2397 {%
2398 \let\glspostlinkhook\relax
2399 \csname#1\expandafter\endcsname\expandafter[\@glxtrp@opt]{#2}[]%
2400 }%
2401 }
```

\@glxtrp

```
2402 \newrobustcmd*{\@glxtrp}[2]{%
2403 \ifcsdef{gls#1}%
2404 {%
2405 \@glxtrp{gls#1}{#2}%
2406 }%
2407 {%
2408 \ifcsdef{glsxtr#1}%
2409 {%
2410 \@glxtrp{glsxtr#1}{#2}%
2411 }%
2412 {%
2413 \PackageError{glossaries-extra}{‘#1’ not recognised by
2414 \string\glxtrp}{}%
2415 }%
2416 }%
2417 }
```

\@Glsxtrp

```
2418 \newrobustcmd*{\@Glsxtrp}[2]{%
```

```

2419 \ifcsdef{Gls#1}%
2420 {%
2421   \@@glxstrp{Gls#1}{#2}%
2422 }%
2423 {%
2424   \ifcsdef{Glsxtr#1}%
2425   {%
2426     \@@glxstrp{Glsxtr#1}{#2}%
2427   }%
2428   {%
2429     \PackageError{glossaries-extra}{‘#1’ not recognised by
2430       \string\Glsxtrp}{}%
2431   }%
2432 }%
2433 }

```

\@GLSxtrp

```

2434 \newrobustcmd*{\@GLSxtrp}[2]{%
2435   \ifcsdef{GLS#1}%
2436   {%
2437     \@@glxstrp{GLS#1}{#2}%
2438   }%
2439   {%
2440     \ifcsdef{GLSxtr#1}%
2441     {%
2442       \@@glxstrp{GLSxtr#1}{#2}%
2443     }%
2444     {%
2445       \PackageError{glossaries-extra}{‘#1’ not recognised by
2446         \string\GLSxtrp}{}%
2447     }%
2448   }%
2449 }

```

\glxstr@entry@p

```

2450 \newrobustcmd*{\glxstr@headentry@p}[2]{%
2451   \gl@ifattribute{#1}{headuc}{true}%
2452   {%
2453     \mfirstucMakeUppercase{\@gl@s@entry@field{#1}{#2}}%
2454   }%
2455   {%
2456     \@gl@s@entry@field{#1}{#2}%
2457   }%
2458 }

```

\glxstrp Not robust as it needs to expand somewhat.

```

2459 \ifdef\texorpdfstring
2460 {
2461   \newcommand{\glxstrp}[2]{%

```

```

2462 \protect\NoCaseChange
2463 {%
2464 \protect\texorpdfstring
2465 {%
2466 \protect\glstrifinmark
2467 {%
2468 \ifcsdef{glstrhead#1}%
2469 {%
2470 {\protect\csuse{glstrhead#1}{#2}}%
2471 }%
2472 {%
2473 \glstr@headentry@p{#2}{#1}%
2474 }%
2475 }%
2476 {%
2477 \@glstrp{#1}{#2}%
2478 }%
2479 }%
2480 {%
2481 \protect\@glstr@entry@field{#2}{#1}%
2482 }%
2483 }%
2484 }
2485 }
2486 {
2487 \newcommand{\glstrp}[2]{%
2488 \protect\NoCaseChange
2489 {%
2490 \protect\glstrifinmark
2491 {%
2492 \ifcsdef{glstrhead#1}%
2493 {%
2494 {\protect\csuse{glstrhead#1}}%
2495 }%
2496 {%
2497 \glstr@headentry@p{#2}{#1}%
2498 }%
2499 }%
2500 {%
2501 \@glstrp{#1}{#2}%
2502 }%
2503 }%
2504 }
2505 }

```

Provide short synonyms for the most common option.

`\glsp`

```

2506 \newcommand*{\glsp}{\glstrp{short}}

```

`\glspt`

```
2507 \newcommand*{\glspt}{\glsxtrp{text}}
```

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

```
2508 \ifdef\texorpdfstring
2509 {
2510   \newcommand{\Glsxtrp}[2]{%
2511     \protect\NoCaseChange
2512     {%
2513       \protect\texorpdfstring
2514       {%
2515         \protect\glsxtrifinmark
2516         {%
2517           \ifcsdef{Glsxtrhead#1}%
2518           {%
2519             {\protect\csuse{Glsxtrhead#1}{#2}}%
2520           }%
2521           {%
2522             \protect\@Gls@entry@field{#2}{#1}%
2523           }%
2524         }%
2525         {%
2526           \@Glsxtrp{#1}{#2}%
2527         }%
2528       }%
2529       {%
2530         \protect\@gls@entry@field{#2}{#1}%
2531       }%
2532     }%
2533   }
2534 }
2535 {
2536   \newcommand{\Glsxtrp}[2]{%
2537     \protect\NoCaseChange
2538     {%
2539       \protect\glsxtrifinmark
2540       {%
2541         \ifcsdef{Glsxtrhead#1}%
2542         {%
2543           {\protect\csuse{Glsxtrhead#1}}%
2544         }%
2545         {%
2546           \protect\@Gls@entry@field{#2}{#1}%
2547         }%
2548       }%
2549       {%
2550         \@Glsxtrp{#1}{#2}%
2551       }%
2552     }%
2553   }
2554 }
```

```

2552     }%
2553   }
2554 }

```

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

```

2555 \ifdef\teorpdfstring
2556 {
2557   \newcommand{\GLSxtrp}[2]{%
2558     \protect\NoCaseChange
2559     {%
2560       \protect\teorpdfstring
2561       {%
2562         \protect\glxtrifinmark
2563         {%
2564           \ifcsdef{GLSxtr#1}%
2565           {%
2566             {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2567           }%
2568           {%
2569             \protect\mfirstucMakeUppercase
2570             {%
2571               \protect\@gls@entry@field{#2}{#1}%
2572             }%
2573           }%
2574         }%
2575         {%
2576           \@GLSxtrp{#1}{#2}%
2577         }%
2578       }%
2579       {%
2580         \protect\@gls@entry@field{#2}{#1}%
2581       }%
2582     }%
2583   }
2584 }
2585 {
2586   \newcommand{\GLSxtrp}[2]{%
2587     \protect\NoCaseChange
2588     {%
2589       \protect\glxtrifinmark
2590       {%
2591         \ifcsdef{GLSxtr#1}%
2592         {%
2593           {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2594         }%
2595         {%
2596           \protect\mfirstucMakeUppercase
2597           {%
2598             \protect\@gls@entry@field{#2}{#1}%

```

```

2599         }%
2600     }%
2601 }%
2602 {%
2603     \@GLSxtrp{#1}{#2}%
2604 }%
2605 }%
2606 }
2607 }

```

1.3.5 Entry Counting

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

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

```

\@glsunset   Global unset.
2608 \renewcommand*{\@glsunset}[1]{%
2609     \@glsunset{#1}%
2610     \glsxtrpostunset{#1}%
2611 }%

glsxtrpostunset
2612 \newcommand*{\glsxtrpostunset}[1]{

\@glslocalunset   Local unset.
2613     \renewcommand*{\@glslocalunset}[1]{%
2614         \@glslocalunset{#1}%
2615         \glsxtrpostlocalunset{#1}%
2616     }%

rpostlocalunset
2617 \newcommand*{\glsxtrpostlocalunset}[1]{

\@glsreset   Global reset.
2618 \renewcommand*{\@glsreset}[1]{%
2619     \@glsreset{#1}%
2620     \glsxtrpostreset{#1}%
2621 }%

glsxtrpostreset
2622 \newcommand*{\glsxtrpostreset}[1]{

\@glslocalreset   Local reset.
2623 \renewcommand*{\@glslocalreset}[1]{%
2624     \@glslocalreset{#1}%
2625     \glsxtrpostlocalreset{#1}%
2626 }%

```

rpostlocalreset

```
2627 \newcommand*{\glxtrpostlocalreset}[1]{}
```

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

```
2628 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

Enable entry counting:

```
2629 \glsenableentrycount
```

Redefine \gls etc:

```
2630 \renewcommand*{\gls}{\cglss}%
2631 \renewcommand*{\Gls}{\cGls}%
2632 \renewcommand*{\glspl}{\cglsspl}%
2633 \renewcommand*{\Glspl}{\cGlspl}%
2634 \renewcommand*{\GLS}{\cGLS}%
2635 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2636 \@glxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
2637 \let\GlsXtrEnableEntryCounting\@glxtr@setentrycountunsetattr
2638 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
2639 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2640 can't be used with \string\GlsXtrEnableEntryCounting}%
2641 {Use one or other but not both commands}}%
2642 }
```

ycountunsetattr

```
2643 \newcommand*{\@glxtr@setentrycountunsetattr}[2]{%
2644 \@for\@glxtr@cat:=#1\do
2645 {%
2646 \ifdefempty{\@glxtr@cat}{}%
2647 {%
2648 \glsssetcategoryattribute{\@glxtr@cat}{entrycount}{#2}%
2649 }%
2650 }%
2651 }
```

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

nableentrycount

```
2652 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
2653 \appto\@newglossaryentry@defcounters{\@@newglossaryentry@defcounters}%
```

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

```
2654 \renewcommand*{\gls@defdocnewglossaryentry}{%
2655 \renewcommand*{\newglossaryentry}[2]{%
```



```

2656     \PackageError{glossaries}{\string\newglossaryentry\space
2657     may only be used in the preamble when entry counting has
2658     been activated}{If you use \string\glsenableentrycount\space
2659     you must place all entry definitions in the preamble not in
2660     the document environment}%
2661   }%
2662 }%

```

New commands to access new fields:

```

2663 \newcommand*{\glsentrycurrcount}[1]{%
2664   \ifcsundef{glo@\glsdetoklabel{##1}@currcount}%
2665   {0}{\@gls@entry@field{##1}{currcount}}}%
2666 }%
2667 \newcommand*{\glsentryprevcount}[1]{%
2668   \ifcsundef{glo@\glsdetoklabel{##1}@prevcount}%
2669   {0}{\@gls@entry@field{##1}{prevcount}}}%
2670 }%

```

Adjust post unset and reset:

```

2671 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
2672 \renewcommand*{\glsxtrpostunset}[1]{%
2673   \@glsxtr@entrycount@org@unset{##1}%
2674   \@gls@increment@currcount{##1}%
2675 }%
2676 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
2677 \renewcommand*{\glsxtrpostlocalunset}[1]{%
2678   \@glsxtr@entrycount@org@localunset{##1}%
2679   \@gls@local@increment@currcount{##1}%
2680 }%
2681 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
2682 \renewcommand*{\glsxtrpostreset}[1]{%
2683   \@glsxtr@entrycount@org@reset{##1}%
2684   \csdef{glo@\glsdetoklabel{##1}@currcount}{0}%
2685 }%
2686 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
2687 \renewcommand*{\glsxtrpostlocalreset}[1]{%
2688   \@glsxtr@entrycount@org@localreset{##1}%
2689   \csdef{glo@\glsdetoklabel{##1}@currcount}{0}%
2690 }%

```

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

```

2691 \let\@cgl@s@\@cgl@s@
2692 \let\@cgl spl@\@cgl spl@

2693 \let\@cGls@\@cGls@
2694 \let\@cGlspl@\@cGlspl@
2695 \let\@cGLS@\@cGLS@
2696 \let\@cGLSpl@\@cGLSpl@

```

The rest is as the original definition.

```

2697 \AtEndDocument{\@gls@write@entrycounts}%
2698 \renewcommand*{\@gls@entry@count}[2]{%
2699   \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
2700 }%
2701 \let\glsenableentrycount\relax
2702 \renewcommand*{\glsenableentryunitcount}{%
2703   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space
2704     can't be used with \string\glsenableentrycount}%
2705   {Use one or other but not both commands}%
2706 }%
2707 }

```

`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.

```

2708 \renewcommand*{\@gls@write@entrycounts}{%
2709   \immediate\write\@auxout
2710   {\string\providecommand*{\string\@gls@entry@count}[2]{}}%
2711   \count@=0\relax
2712   \forallglsentries{\@glsentry}{%
2713     \glshasattribute{\@glsentry}{entrycount}%
2714     {%
2715       \ifglused{\@glsentry}%
2716       {%
2717         \immediate\write\@auxout
2718         {\string\@gls@entry@count{\@glsentry}{\glentrycurrcount{\@glsentry}}}%
2719       }%
2720     }%
2721     \advance\count@ by \@ne
2722   }%
2723 }%
2724 }%
2725 \ifnum\count@=0
2726   \GlossariesExtraWarningNoLine{Entry counting has been enabled
2727     \MessageBreak with \string\glsenableentrycount\space but the
2728     \MessageBreak attribute 'entrycount' hasn't
2729     \MessageBreak been assigned to any of the defined
2730     \MessageBreak entries}%
2731 \fi
2732 }

```

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

```

2733 \newcommand*{\glxtrifcounttrigger}[3]{%
2734   \glshasattribute{#1}{entrycount}%
2735   {%
2736     \ifnum\glentryprevcount{#1}>\glsetattribute{#1}{entrycount}\relax

```

```

2737     #3%
2738   \else
2739     #2%
2740   \fi
2741 }%
2742 {#3}%
2743 }

```

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

\@@cgl@

```

2744 \def\@@cgl@#1#2[#3]{%
2745   \glxtrifcounttrigger{#2}%
2746   {%
2747     \cglformat{#2}{#3}%
2748     \glunset{#2}%
2749   }%
2750   {%
2751     \@gls@{#1}{#2}[#3]%
2752   }%
2753 }%

```

\@@cgl@

```

2754 \def\@@cgl@#1#2[#3]{%
2755   \glxtrifcounttrigger{#2}%
2756   {%
2757     \cglformat{#2}{#3}%
2758     \glunset{#2}%
2759   }%
2760   {%
2761     \@gls@{#1}{#2}[#3]%
2762   }%
2763 }%

```

\@@cGls@

```

2764 \def\@@cGls@#1#2[#3]{%
2765   \glxtrifcounttrigger{#2}%
2766   {%
2767     \cGlsformat{#2}{#3}%
2768     \glunset{#2}%
2769   }%
2770   {%
2771     \@Gls@{#1}{#2}[#3]%
2772   }%
2773 }%

```

\@@cGlspl@

```

2774 \def\@@cGlspl@#1#2[#3]{%
2775   \glxtrifcounttrigger{#2}%

```

```

2776 {%
2777   \cGlsplformat{#2}{#3}%
2778   \glsunset{#2}%
2779 }%
2780 {%
2781   \@Glspl@{#1}{#2}[#3]%
2782 }%
2783 }%

```

\@@cGLS@

```

2784 \def\@@cGLS@#1#2[#3]{%
2785   \glsxtrifcounttrigger{#2}%
2786   {%
2787     \cGLSformat{#2}{#3}%
2788     \glsunset{#2}%
2789   }%
2790   {%
2791     \@GLS@{#1}{#2}[#3]%
2792   }%
2793 }%

```

\@@cGLSpl@

```

2794 \def\@@cGLSpl@#1#2[#3]{%
2795   \glsxtrifcounttrigger{#2}%
2796   {%
2797     \cGLSplformat{#2}{#3}%
2798     \glsunset{#2}%
2799   }%
2800   {%
2801     \@GLSpl@{#1}{#2}[#3]%
2802   }%
2803 }%

```

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

\@cgl's@

```

2804 \def\@cgl's@#1#2[#3]{\@gls@{#1}{#2}[#3]}

```

\@cGls@

```

2805 \def\@cGls@#1#2[#3]{\@Gls@{#1}{#2}[#3]}

```

\@cgl'spl@

```

2806 \def\@cgl'spl@#1#2[#3]{\@glspl@{#1}{#2}[#3]}

```

\@cGlspl@

```

2807 \def\@cGlspl@#1#2[#3]{\@Glspl@{#1}{#2}[#3]}

```

Add all upper case versions not provided by glossaries.

\cGLS

2808 \newrobustcmd*{\cGLS}{\@gls@hyp@opt\@cGLS}

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

2809 \newcommand*{\@cGLS}[2][\%

2810 \new@ifnextchar[\@cGLS@{#1}{#2}]{\@cGLS@{#1}{#2}[]}%

2811 }

\@cGLS@

2812 \def\@cGLS@#1#2[#3]{\@cGLS@{#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.

2813 \newcommand*{\cGLSformat}[2]{%

2814 \expandafter\mfirstucMakeUppercase\expandafter{\cglformat{#1}{#2}}%

2815 }

\cGLSpl

2816 \newrobustcmd*{\cGLSpl}{\@gls@hyp@opt\@cGLSpl}

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

2817 \newcommand*{\@cGLSpl}[2][\%

2818 \new@ifnextchar[\@cGLSpl@{#1}{#2}]{\@cGLSpl@{#1}{#2}[]}%

2819 }

\@cGLSpl@

2820 \def\@cGLSpl@#1#2[#3]{\@cGLSpl@{#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.

2821 \newcommand*{\cGLSplformat}[2]{%

2822 \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%

2823 }

Modify the trigger formats to check for the regular attribute.

\cglformat

2824 \renewcommand*{\cglformat}[2]{%

2825 \glsifregular{#1}

2826 {\glsentryfirst{#1}}%

2827 {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%

2828 }

\cGlsformat

2829 \renewcommand*{\cGlsformat}[2]{%

2830 \glsifregular{#1}

2831 {\Glsentryfirst{#1}}%

2832 {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%

2833 }

\cglsplformat

```
2834 \renewcommand*{\cglsplformat}[2]{%
2835   \glsifregular{#1}
2836   {\glsentryfirstplural{#1}}%
2837   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}\#2%
2838 }
```

\cGlsplformat

```
2839 \renewcommand*{\cGlsplformat}[2]{%
2840   \glsifregular{#1}
2841   {\Glsentryfirstplural{#1}}%
2842   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}}\#2%
2843 }
```

New code similar to above for unit counting.

defunitcounters

```
2844 \newcommand*{\@newglossaryentry@defunitcounters}{%
2845   \edef\@glo@countunit{\csuse{@glxtr@categoryattr@{\@glo@category @unitcount}}}%
2846   \ifdefvoid\@glo@countunit
2847   {}%
2848   {%
2849     \@glxtr@ifunitcounter{\@glo@countunit}%
2850     {}%
2851     {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}}%
2852   }%
2853 }
```

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

```
2854 \newcommand*{\@glxtr@unitcountlist}{}
```

@addunitcounter

```
2855 \newcommand*{\@glxtr@addunitcounter}[1]{%
2856   \listadd{\@glxtr@unitcountlist}{#1}%
2857   \ifcsundef{glxtr@theunit@#1}
2858   {%
2859     \ifcsdef{theH#1}%
2860     {\csdef{glxtr@theunit@#1}{\csuse{theH#1}}}%
2861     {\csdef{glxtr@theunit@#1}{\csuse{the#1}}}%
2862   }%
2863   {}%
2864 }
```

r@ifunitcounter

```
2865 \newcommand*{\@glxtr@ifunitcounter}[3]{%
2866   \xifinlist{#1}{\@glxtr@unitcountlist}{#2}{#3}%
2867 }
```

urrentunitcount

```
2868 \newcommand*\@glxtr@currentunitcount[1]{%
2869   glo@\glstoklabel{#1}@currunit@\glstgetattribute{#1}{unitcount}.%
2870   \csuse{glxtr@theunit@\glstgetattribute{#1}{unitcount}}%
2871 }
```

previousunitcount

```
2872 \newcommand*\@glxtr@previousunitcount[1]{%
2873   glo@\glstoklabel{#1}@prevunit@\glstgetattribute{#1}{unitcount}.%
2874   \csuse{glxtr@theunit@\glstgetattribute{#1}{unitcount}}%
2875 }
```

t@currunitcount

```
2876 \newcommand*\@glst@increment@currunitcount[1]{%
2877   \glshasattribute{#1}{unitcount}%
2878   {%
2879     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
2880     \ifcsundef{\@glxtr@csname}%
2881     {%
2882       \csgdef{\@glxtr@csname}{1}%
2883       \listcsxadd
2884         {glo@\glstoklabel{#1}@unitlist}%
2885         {\glstgetattribute{#1}{unitcount}.%
2886         \csuse{glxtr@theunit@\glstgetattribute{#1}{unitcount}}%
2887         }%
2888     }%
2889     {%
2890       \csxdef{\@glxtr@csname}%
2891       {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
2892     }%
2893   }%
2894   {%
2895 }
```

t@currunitcount

```
2896 \newcommand*\@glst@local@increment@currunitcount[1]{%
2897   \glshasattribute{#1}{unitcount}%
2898   {%
2899     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
2900     \ifcsundef{\@glxtr@csname}%
2901     {%
2902       \csdef{\@glxtr@csname}{1}%
2903       \listcseadd
2904         {glo@\glstoklabel{#1}@unitlist}%
2905         {\glstgetattribute{#1}{unitcount}.%
2906         \csuse{glxtr@theunit@\glstgetattribute{#1}{unitcount}}%
2907         }%
2908     }%
2909     {%
```

```

2910      \csedef{\@glsxtr@csname}%
2911      {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
2912      }%
2913      }%
2914      {}%
2915      }

```

r@currunitcount

```

2916 \newcommand*{\@glsxtr@currunitcount}[2]{%
2917   \ifcsundef
2918     {glo@\glsdetoklabel{#1}@currunit@#2}%
2919     {0}%
2920     {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}%
2921   }%

```

r@prevunitcount

```

2922 \newcommand*{\@glsxtr@prevunitcount}[2]{%
2923   \ifcsundef
2924     {glo@\glsdetoklabel{#1}@prevunit@#2}%
2925     {0}%
2926     {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}%
2927   }%

```

eentryunitcount

```

2928 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
2929   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
2930   \renewcommand*{\gls@defdocnewglossaryentry}{%
2931     \renewcommand*{\newglossaryentry}[2]{%
2932       \PackageError{glossaries}{\string\newglossaryentry\space
2933       may only be used in the preamble when entry counting has
2934       been activated}{If you use \string\glsenableentryunitcount\space
2935       you must place all entry definitions in the preamble not in
2936       the document environment}%
2937     }%
2938   }%

```

New commands to access new fields:

```

2939   \newcommand*{\glsentrycurrcount}[1]{%
2940     \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2941     \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2942   }%
2943   \newcommand*{\glsentryprevcount}[1]{%
2944     \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
2945     \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
2946   }%

```


Access total count:

```
2947 \newcommand*{\glsentryprevtotalcount}[1]{%
2948   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
2949     {0}%
2950     {%
2951       \number\csuse{glo@\glsdetoklabel{##1}@prevunittotal}
2952     }%
2953   }%
```

Access max value:

```
2954 \newcommand*{\glsentryprevmaxcount}[1]{%
2955   \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
2956     {0}%
2957     {%
2958       \number\csuse{glo@\glsdetoklabel{##1}@prevunitmax}
2959     }%
2960   }%
```

Adjust post unset and reset:

```
2961 \let\@glsxtr@entryunitcount@org@unset\glsxtrpostunset
2962 \renewcommand*{\glsxtrpostunset}[1]{%
2963   \@glsxtr@entryunitcount@org@unset{##1}%
2964   \@gls@increment@currunitcount{##1}%
2965 }%
2966 \let\@glsxtr@entryunitcount@org@localunset\glsxtrpostlocalunset
2967 \renewcommand*{\glsxtrpostlocalunset}[1]{%
2968   \@glsxtr@entryunitcount@org@localunset{##1}%
2969   \@gls@local@increment@currunitcount{##1}%
2970 }%
2971 \let\@glsxtr@entryunitcount@org@reset\glsxtrpostreset
2972 \renewcommand*{\glsxtrpostreset}[1]{%
2973   \glshasattribute{##1}{unitcount}%
2974   {%
2975     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{##1}}%
2976     \ifcsundef{\@glsxtr@csname}%
2977       {}%
2978       {\csgdef{\@glsxtr@csname}{0}}%
2979     }%
2980     {}%
2981   }%
2982 \let\@glsxtr@entryunitcount@org@localreset\glsxtrpostlocalreset
2983 \renewcommand*{\glsxtrpostlocalreset}[1]{%
2984   \@glsxtr@entryunitcount@org@localreset{##1}%
2985   \glshasattribute{##1}{unitcount}%
2986   {%
2987     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{##1}}%
2988     \ifcsundef{\@glsxtr@csname}%
2989       {}%
2990       {\csdef{\@glsxtr@csname}{0}}%
2991   }%
```

```

2992    {}%
2993  }%

```

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

```

2994  \let\@cgls@\@cgls@
2995  \let\@cglspl@\@cglspl@

2996  \let\@cGls@\@cGls@
2997  \let\@cGlspl@\@cGlspl@
2998  \let\@cGLS@\@cGLS@
2999  \let\@cGLSpl@\@cGLSpl@

```

Write information to the aux file.

```

3000  \AtEndDocument{\@gls@write@entryunitcounts}%
3001  \renewcommand*{\@gls@entry@unitcount}[3]{%
3002    \csgdef{glo@glstdetoklabel{##1}@prevunit@##3}{##2}%
3003    \ifcsundef{glo@glstdetoklabel{##1}@prevunittotal}%
3004    {\csgdef{glo@glstdetoklabel{##1}@prevunittotal}{##2}}%
3005    {%
3006      \csxdef{glo@glstdetoklabel{##1}@prevunittotal}{
3007        \number\numexpr\csuse{glo@glstdetoklabel{##1}@prevunittotal}+##2}%
3008      }%
3009      \ifcsundef{glo@glstdetoklabel{##1}@prevunitmax}%
3010      {\csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}}%
3011      {%
3012        \ifnum\csuse{glo@glstdetoklabel{##1}@prevunitmax}<##2
3013        \csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}%
3014        \fi
3015      }%
3016    }%
3017  \let\glsenableentryunitcount\relax
3018  \renewcommand*{\glsenableentrycount}{%
3019    \PackageError{glossaries-extra}{\string\glsenableentrycount\space
3020      can't be used with \string\glsenableentryunitcount}%
3021    {Use one or other but not both commands}%
3022  }%
3023 }
3024 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

3025 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

3026 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
3027   \immediate\write\@auxout
3028   {\string\@gls@entry@unitcount
3029     {\@glsentry}%
3030     {\@glsxtr@currunitcount{\@glsentry}{##1}%
3031     }%

```

```

3032     {#1}}}%
3033 }

```

entryunitcounts

```

3034 \newcommand*{\@gls@write@entryunitcounts}{%
3035   \immediate\write\@auxout
3036     {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}}%
3037   \count@=0\relax
3038   \forallglsentries{\@glsentry}{%
3039     \glshasattribute{\@glsentry}{unitcount}%
3040     {%
3041       \ifglsused{\@glsentry}%
3042       {%
3043         \forlistcsloop
3044           {\@gls@write@entryunitcounts@do}%
3045           {glo@\glsdetoklabel{\@glsentry}@unitlist}%
3046       }%
3047     }%
3048     \advance\count@ by \@ne
3049   }%
3050 }%
3051 }%
3052 \ifnum\count@=0
3053   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3054     \MessageBreak with \string\glsenableentryunitcount\space but the
3055     \MessageBreak attribute ‘unitcount’ hasn’t
3056     \MessageBreak been assigned to any of the defined
3057     \MessageBreak entries}%
3058 \fi
3059 }

```

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

```

3060 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

3061   \glsenableentryunitcount

```

Redefine \gls etc:

```

3062   \renewcommand*{\gls}{\cgl}%
3063   \renewcommand*{\Gls}{\cGls}%
3064   \renewcommand*{\glspl}{\cglspl}%
3065   \renewcommand*{\Glspl}{\cGlspl}%
3066   \renewcommand*{\GLS}{\cGLS}%
3067   \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

3068   \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

3069 \let\GlsXtrEnableEntryUnitCounting\@glxtr@setentryunitcountunsetattr
3070 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
3071 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
3072 can't be used with \string\GlsXtrEnableEntryUnitCounting}%
3073 {Use one or other but not both commands}}%
3074 }

```

tcountunsetattr

```

3075 \newcommand*{\@glxtr@setentryunitcountunsetattr}[3]{%
3076 \@for\@glxtr@cat:=#1\do
3077 {%
3078 \ifdefempty{\@glxtr@cat}{}%
3079 {%
3080 \glsssetcategoryattribute{\@glxtr@cat}{entrycount}{#2}%
3081 \glsssetcategoryattribute{\@glxtr@cat}{unitcount}{#3}%
3082 }%
3083 }%
3084 }

```

1.3.6 Acronym Modifications

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

nericNewAcronym

```

3085 \renewcommand*{\SetGenericNewAcronym}{%
3086 \let\@Gls@entryname\@Gls@acentryname
3087 \renewcommand{\newacronym}[4][{}]{%
3088 \ifdefempty{\@glsacronymlists}%
3089 {%
3090 \def\@glo@type{\acronymtype}%
3091 \setkeys{glossentry}{##1}%
3092 \DeclareAcronymList{\@glo@type}%
3093 }%
3094 }%
3095 \glskeylisttok{##1}%
3096 \glslabeltok{##2}%
3097 \glsshorttok{##3}%
3098 \glslongtok{##4}%
3099 \newacronymhook
3100 \protected@edef\@do@newglossaryentry{%
3101 \noexpand\newglossaryentry{\the\glslabeltok}%
3102 {%
3103 type=\acronymtype,%

```

```

3104     name={\expandonce{\acronymentry{##2}}},%
3105     sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
3106     text={\the\glsshorttok},%
3107     short={\the\glsshorttok},%
3108     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3109     long={\the\glslongtok},%
3110     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3111     category=acronym,
3112     \GenericAcronymFields,%
3113     \the\glskeylisttok
3114 }%
3115 }%
3116 \do@newglossaryentry
3117 }%
3118 \renewcommand*{\acrfullfmt}[3]{%
3119   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
3120 \renewcommand*{\Acrfullfmt}[3]{%
3121   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
3122 \renewcommand*{\ACRfullfmt}[3]{%
3123   \glslink[##1]{##2}{%
3124     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
3125 \renewcommand*{\acrfullplfmt}[3]{%
3126   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
3127 \renewcommand*{\Acrfullplfmt}[3]{%
3128   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
3129 \renewcommand*{\ACRfullplfmt}[3]{%
3130   \glslink[##1]{##2}{%
3131     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
3132 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}}%
3133 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
3134 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}}%
3135 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
3136 }

```

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:

```

3137 \let\@glxtr@org@setacronymstyle\setacronymstyle
3138 \let\@glxtr@org@newacronymstyle\newacronymstyle

```

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

```

3139 \newcommand*{\MakeAcronymsAbbreviations}{%
3140   \renewcommand*{\newacronym}[4][{}]{%
3141     \glxtr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
3142   }%
3143   \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
3144   \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%

```

```

3145 \renewcommand*\setacronymstyle}[1]{%
3146 \PackageError{glossaries-extra}{\string\setacronymstyle{##1}
3147 unavailable.
3148 Use \string\setabbreviationstyle\space instead.
3149 The original acronym interface can be restored with
3150 \string\RestoreAcronyms}{}%
3151 }%
3152 \renewcommand*\newacronymstyle}[1]{%
3153 \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
3154 available unless you restore the original acronym interface with
3155 \string\RestoreAcronyms}%
3156 \@glsxtr@org@newacronymstyle{##1}%
3157 }%
3158 }

```

Switch acronyms to abbreviations:

```
3159 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```

3160 \newcommand*\RestoreAcronyms){%
3161 \SetGenericNewAcronym
3162 \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
3163 \renewcommand{\acronymfont}[1]{##1}%
3164 \let\setacronymstyle\@glsxtr@org@setacronymstyle
3165 \let\newacronymstyle\@glsxtr@org@newacronymstyle

```

Need to restore the original definition of `\@gls@link@checkfirsthyper` but `\glsxtrifwasfirstuse` still needs setting for the benefit of the post-link hook.

```

3166 \renewcommand*\@gls@link@checkfirsthyper{%
3167 \ifglsused{\glslabel}%
3168 {\let\glsxtrifwasfirstuse\@secondoftwo}
3169 {\let\glsxtrifwasfirstuse\@firstoftwo}%
3170 \@glsxtr@org@checkfirsthyper
3171 }
3172 \glssetcategoryattribute{acronym}{regular}{false}%
3173 \setacronymstyle{long-short}%
3174 }

```

\glsacspace Allow the user to customise the maximum value.

```

3175 \renewcommand*\glsacspace}[1]{%
3176 \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
3177 \ifdim\dimen@<\glsacspacemax~\else\space\fi
3178 }

```

\glsacspacemax Value used in the above.

```
3179 \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

```
3180 \newcommand*{\@glxtr@reg@glosslist}{{}
```

Save the original definition of `\makeglossaries`:

```
3181 \let\@glxtr@org@makeglossaries\makeglossaries
```

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

\makeglossaries

```
3182 \renewcommand*{\makeglossaries}[1] [] {%
3183   \ifblank{#1}%
3184   {\@glxtr@org@makeglossaries}%
3185   {%
3186     \edef\@glxtr@reg@glosslist{#1}%
3187     \ifundef{\glswrite}{\newwrite\glswrite}{}%
3188     \protected@write\@auxout{}{\string\providecommand
3189       \string\@glsorder[1]}{}
3190     \protected@write\@auxout{}{\string\providecommand
3191       \string\@istfilename[1]}{}
3192     \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
3193     \protected@write\@auxout{}{\string\@glsorder{\glsorder}}
3194     \protected@write\@auxout{}{\string\glxtr@makeglossaries{#1}}
3195     \write\@auxout{\string\providecommand\string\@gls@reference[3]}{}}%
```

Iterate through each supplied glossary type and activate it.

```
3196   \@for\@glo@type:=#1\do{%
3197     \ifdefempty{\@glo@type}{}{\@makeglossary{\@glo@type}}%
3198   }%
```

New glossaries must be created before `\makeglossaries`:

```
3199   \renewcommand*\newglossary[4] [] {%
3200     \PackageError{glossaries}{New glossaries
3201       must be created before \string\makeglossaries}{You need
3202       to move \string\makeglossaries\space after all your
3203       \string\newglossary\space commands}}%
```

Any subsequence instances of this command should have no effect

```
3204   \let\@makeglossary\relax
3205   \let\makeglossary\relax
3206   \renewcommand\makeglossaries[1] [] {}%
```

Disable all commands that have no effect after \makeglossaries

```
3207 \let\@disable@onlypremakeg
```

Allow see key:

```
3208 \let\gls@checkseeallowed\relax
```

Adjust \do@seeglossary

```
3209 \renewcommand*\do@seeglossary}[2]{%
3210 \edef\@gls@label{\glsdetoklabel{##1}}%
3211 \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3212 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3213 {\@glsxtr@org@doseeglossary{##1}{##2}}%
3214 {%
3215 \protected@write\@auxout{}{%
3216 \string\@gls@reference
3217 {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
3218 }%
3219 }%
3220 }%
```

Adjust \do@wrglossary

```
3221 \let\@glsxtr@do@wrglossary\do@wrglossary
3222 \def\do@wrglossary{%
3223 \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3224 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3225 {\@glsxtr@do@wrglossary}%
3226 {\gls@noidxglossary}%
3227 }%
```

Suppress warning about no \makeglossaries

```
3228 \let\warn@nomakeglossaries\relax
3229 \def\warn@noprintglossary{%
3230 \GlossariesWarningNoLine{No \string\printglossary\space
3231 or \string\printglossaries\space
3232 found.^^J(Remove \string\makeglossaries\space if you don't want
3233 any glossaries.)^^JThis document will not have a glossary}%
3234 }%
```

Only warn for glossaries not listed.

```
3235 \renewcommand{\@gls@noref@warn}[1]{%
3236 \edef\@gls@type{##1}%
3237 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3238 {%
3239 \GlossariesExtraWarning{Can't use
3240 \string\printnoidxglossary[type={\@gls@type}]
3241 when '\@gls@type' is listed in the optional argument of
3242 \string\makeglossaries}%
3243 }%
3244 {%
3245 \GlossariesWarning{Empty glossary for
3246 \string\printnoidxglossary[type={##1}].
```



```

3247      Rerun may be required (or you may have forgotten to use
3248      commands like \string\gls)}%
3249  }%
3250 }%

```

Adjust display number list to check for type:

```

3251 \renewcommand*\glsdisplaynumberlist[1]{%
3252   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3253   {\@glsxtr@idx@displaynumberlist{##1}}%
3254   {\@glsxtr@noidx@displaynumberlist{##1}}%
3255 }%

```

Adjust entry list:

```

3256 \renewcommand*\glsentrynumberlist[1]{%
3257   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3258   {\@glsxtr@idx@entrynumberlist{##1}}%
3259   {\@glsxtr@noidx@entrynumberlist{##1}}%
3260 }%

```

Adjust number list loop

```

3261 \renewcommand*\glsnumberlistloop[2]{%
3262   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3263   {%
3264     \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3265     not available for glossary ‘##1’}{}%
3266   }%
3267   {\@glsxtr@noidx@numberlistloop{##1}{##2}}%
3268 }%

```

Only sanitize sort for normal indexing glossaries.

```

3269 \renewcommand*\glsprestandardsort[3]{%
3270   \expandafter\DTLifinlist\expandafter{##2}{\@glsxtr@reg@glosslist}%
3271   {%
3272     \glsdosanitizesort
3273   }%
3274   {%
3275     \ifglssanitizesort
3276       \@gls@noidx@sanitizesort
3277     \else
3278       \@gls@noidx@nosanitizesort
3279     \fi
3280   }%
3281 }%

```

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

```

3282 \renewcommand*\new@glossaryentry[2]{%
3283   \PackageError{glossaries-extra}{Glossary entries must be defined
3284   in the preamble\MessageBreak when you use the optional argument
3285   of \string\makeglossaries}{Either move your definitions to the
3286   preamble or don't use the optional argument of

```

```

3287 \string\makeglossaries}%
3288 }%

```

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

```

3289 \let\@glo@assign@sortkey\@glstr@mixed@assign@sortkey
3290 \renewcommand*\@printgloss@setsort}{%

```

Need to extract just the type value.

```

3291 \expandafter\@glstr@gettype\expandafter,\@glstr@printglossopts,%
3292 type=\glsdefaulttype,\@end@glstr@gettype
3293 \def\@glo@sorttype{\@glo@default@sorttype}%
3294 }%

```

Check automake setting:

```

3295 \ifglsautomake
3296 \renewcommand*\@gls@doautomake}{%
3297 \for\@gls@type:=\@glstr@reg@glosslist\do{%
3298 \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3299 }%
3300 }%
3301 \fi

```

Check the sort setting (glossaries v4.30 onwards):

```

3302 \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
3303 }%
3304 }

```

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

`\orgprintglossary` This no longer simply saves `\@printglossary` with `\let` is actually defined to check for the existence of the title, since ignored glossaries don't have a title assigned. (bib2gls writes `\provideignoredglossary` to the `glstex` file for some settings, so the glossary might not have been defined.) (This command is also used for on-the-fly setting.)

```

3305 \newcommand{\@glstr@orgprintglossary}[2]{%
3306 \def\@glo@type{\glsdefaulttype}%

```

Add check here.

```

3307 \def\glossarytitle{%
3308 \ifcsdef{\@glo@type \@glo@type @title}%
3309 {\csuse{\@glo@type \@glo@type @title}}%
3310 {\glossaryname}}%
3311 \def\glossarytoctitle{\glossarytitle}%
3312 \let\org@glossarytitle\glossarytitle
3313 \def\@glossarystyle{%
3314 \ifx\@glossary@default@style\relax
3315 \GlossariesWarning{No default glossary style provided \MessageBreak
3316 for the glossary '\@glo@type'. \MessageBreak
3317 Using deprecated fallback. \MessageBreak
3318 To fix this set the style with \MessageBreak

```

```

3319      \string\setglossarystyle\space or use the \MessageBreak
3320      style key=value option}%
3321  \fi
3322 }%
3323 \def\gls@dotocitle{\gls@dotocitle{\@glo@type}}%
3324 \let\org@glossaryentrynumbers\glossaryentrynumbers
3325 \bgroup
3326   \@printgloss@setsort
3327   \setkeys{printgloss}{#1}%
3328   \ifx\glossarytitle\org@glossarytitle
3329   \else
3330     \cslet{\glo@type@\@glo@type @title}{\glossarytitle}%
3331   \fi
3332   \let\currentglossary\@glo@type
3333   \let\org@glossaryentrynumbers\glossaryentrynumbers
3334   \let\glsnonextpages\glsnonextpages
3335   \let\glsnextpages\glsnextpages
3336   \let\nopostdesc\@nopostdesc
3337   \gls@dotocitle
3338   \@glossarystyle
3339   \let\gls@org@glossaryentryfield\glossentry
3340   \let\gls@org@glossarysubentryfield\subglossentry
3341   \renewcommand{\glossentry}[1]{%
3342     \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3343     \gls@org@glossaryentryfield{##1}%
3344   }%
3345   \renewcommand{\subglossentry}[2]{%
3346     \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3347     \gls@org@glossarysubentryfield{##1}{##2}%
3348   }%
3349   \@gls@preglossaryhook
3350   #2%
3351 \egroup
3352 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
3353 \global\let\warn@noprintglossary\relax
3354 }

```

\@printglossary Redefine.

```

3355 \renewcommand{\@printglossary}[2]{%
3356   \def\@glsxtr@printglossopts{#1}%
3357   \@glsxtr@orgprintglossary{#1}{#2}%
3358 }

```

Add a key that switches off the entry targets:

```

3359 \define@choicekey{printgloss}{target}[\val\nr]{true,false}[true]{%
3360   \ifcase\nr
3361     \let\@gls@target\glsdohypertarget
3362   \else
3363     \let\@gls@target\@secondoftwo

```

```

3364 \fi
3365 }

```

@makeglossaries For the benefit of makeglossaries

```

3366 \newcommand*{\glxtr@makeglossaries}[1]{%

```

@glxtr@gettype Get just the type.

```

3367 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
3368 \def\@glo@type{#2}%
3369 }

```

@assign@sortkey Assign the sort key.

```

3370 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
3371 \edef\@glo@type{\@glo@type}%
3372 \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
3373 {%
3374 \@glo@no@assign@sortkey{#1}%
3375 }%
3376 {%
3377 \@glo@assign@sortkey{#1}%
3378 }%
3379 }%

```

Display number list for the regular version:

splaynumberlist

```

3380 \let\@glxtr@idx@displaynumberlist\glsdisplaynumberlist

```

Display number list for the “noidx” version:

splaynumberlist

```

3381 \newcommand*{\@glxtr@noidx@displaynumberlist}[1]{%
3382 \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3383 \ifdef\@gls@loclist
3384 {%
3385 \def\@gls@noidxloclist@sep{%
3386 \def\@gls@noidxloclist@sep{%
3387 \def\@gls@noidxloclist@sep{%
3388 \glsnumlistsep
3389 }%
3390 \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
3391 }%
3392 }%
3393 \def\@gls@noidxloclist@finalsep{}%
3394 \def\@gls@noidxloclist@prev{}%
3395 \forlistloop{\@gls@noidxdisplayloclisthandler}{\@gls@loclist}%
3396 \@gls@noidxloclist@finalsep
3397 \@gls@noidxloclist@prev
3398 }%
3399 {%

```

```

3400 \glstrundef\tag
3401 \glsdoifexists{#1}%
3402 {%
3403     \GlossariesWarning{Missing location list for ‘#1’. Either
3404         a rerun is required or you haven’t referenced the entry.}%
3405 }%
3406 }%
3407 }%
3408

```

And for the number list loop:

@numberlistloop

```

3409 \newcommand*{\@glstr@noidx@numberlistloop}[3]{%
3410 \letcs{\@gls@loclist}{glo@glstoklabel{#1}@loclist}%
3411 \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
3412 \let\@gls@org@glssseeformat\glssseeformat
3413 \let\glsnoidxdisplayloc#2\relax
3414 \let\glssseeformat#3\relax
3415 \ifdef\@gls@loclist
3416 {%
3417     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3418 }%
3419 {%

3420 \glstrundef\tag
3421 \glsdoifexists{#1}%
3422 {%
3423     \GlossariesWarning{Missing location list for ‘##1’. Either
3424         a rerun is required or you haven’t referenced the entry.}%
3425 }%
3426 }%
3427 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3428 \let\glssseeformat\@gls@org@glssseeformat
3429 }%

```

Same for entry number list.

entrynumberlist

```

3430 \newcommand*{\@glstr@noidx@entrynumberlist}[1]{%
3431 \letcs{\@gls@loclist}{glo@glstoklabel{#1}@loclist}%
3432 \ifdef\@gls@loclist
3433 {%
3434     \glsnoidxloclist{\@gls@loclist}%
3435 }%
3436 {%

3437 \glstrundef\tag
3438 \glsdoifexists{#1}%
3439 {%
3440     \GlossariesWarning{Missing location list for ‘#1’. Either

```

```

3441         a rerun is required or you haven't referenced the entry.}%
3442     }%
3443 }%
3444 }%

```

entrynumberlist

```

3445 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

3446 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
3447   \protected@edef\@glsxtr@titlelabel{#1}%
3448   \ifdefvoid\@glsxtr@titlelabel
3449     {}%
3450     {%
3451       \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle@#1}}%
3452     }%
3453   \ifdefvoid{\@glsxtr@titlelabel}%
3454     {%
3455       \DTLifint{#1}%
3456       {%
3457         \ifnum#1<256\relax
3458           \edef#2{\char#1\relax}%
3459         \else
3460           \edef#2{#1}%
3461         \fi
3462       }%
3463     {%
3464       \ifcsundef{#1groupname}%
3465         {\def#2{#1}}%
3466         {\letcs#2{#1groupname}}%
3467     }%
3468   }%
3469   {%
3470     \let#2\@glsxtr@titlelabel
3471   }%
3472 }

```

g@getgrouptitle Save original definition of \@gls@getgrouptitle

```

3473 \let@glsxtr@org@getgrouptitle\@gls@getgrouptitle

```

trgetgrouptitle Provide a user-level command to fetch the group title. The first argument is the group label.

The second argument is a control sequence in which to store the title.

```

3474 \newrobustcmd{\glsxtrgetgrouptitle}[2]{%
3475   \protected@edef\@glsxtr@titlelabel{glsxtr@grouptitle@#1}%
3476   \@onelevel@sanitize\@glsxtr@titlelabel
3477   \ifcsdef{\@glsxtr@titlelabel}
3478     {\letcs{#2}{\@glsxtr@titlelabel}}%
3479     {\glsxtr@org@getgrouptitle{#1}{#2}}%
3480 }

```

```
3481 \let\@gls@getgrouptitle\glsxtrgetgrouptitle
```

`\trsetgrouptitle` Sets the title for the given group label.

```
3482 \newcommand{\glsxtrsetgrouptitle}[2]{%
3483   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3484   \@onelevel@sanitize\@glsxtr@titlelabel
3485   \csxdef{\@glsxtr@titlelabel}{#2}%
3486 }
```

`\glsnavigation` Redefine to use new user-level command.

```
3487 \renewcommand*{\glsnavigation}{%
3488   \def\@gls@between{}%
3489   \ifcsundef{\@gls@hypergrouplist@\@glo@type}%
3490   {%
3491     \def\@gls@list{}%
3492   }%
3493   {%
3494     \expandafter\let\expandafter\@gls@list
3495     \csname \@gls@hypergrouplist@\@glo@type\endcsname
3496   }%
3497   \@for\@gls@tmp:=\@gls@list\do{%
3498     \@gls@between
3499     \glsxtrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
3500     \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
3501     \let\@gls@between\glshypernavsep
3502   }%
3503 }
```

`@noidx@glossary`

```
3504 \renewcommand*{\@print@noidx@glossary}{%
3505   \ifcsdef{\@gls@ref@\@glo@type}%
3506   {%
3507     \ifcsdef{\@glo@sortmacro@\@glo@sorttype}%
3508     {%
3509       \csuse{\@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
3510     }%
3511     {%
3512       \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
3513     }%
3514     \glossarysection[\glossarytoctitle]{\glossarytitle}%
3515     \glossarypreamble
```

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

```
3516   \def\@gls@currentlettergroup{}%
3517   \begin{theglossary}%
3518   \glossaryheader
3519   \glsresetentrylist
3520   \forlistcsloop{\@gls@noidx@do}{\@gls@ref@\@glo@type}%
```

```

3521 \end{theglossary}%
3522 \glossarypostamble
3523 }%
3524 {%

```

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

```

3525 \glxtrifemptyglossary{\@glo@type}%
3526 }%
3527 {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
3528 \@gls@noref@warn{\@glo@type}%
3529 }%
3530 }

```

`noidxdisplayloc` Patch to check for range formations.

```

3531 \renewcommand*{\glsnoidxdisplayloc}[4]{%
3532 \setentrycounter[#1]{#2}%
3533 \@glxtr@display@loc#3\empty\end@glxtr@display@loc{#4}%
3534 }

```

`xtr@display@loc` Patch to check for range formations.

```

3535 \def\@glxtr@display@loc#1#2\end@glxtr@display@loc#3{%
3536 \ifx#1(\relax
3537 \glxtrdisplaystartloc{#2}{#3}%
3538 \else
3539 \ifx#1)\relax
3540 \glxtrdisplayendloc{#2}{#3}%
3541 \else
3542 \glxtrdisplaysingleloc{#1#2}{#3}%
3543 \fi
3544 \fi
3545 }

```

`isplaysingleloc` Single location.

```

3546 \newcommand*{\glxtrdisplaysingleloc}[2]{%
3547 \csuse{#1}{#2}%
3548 }

```

By default the range identifiers are simply ignored. A custom list loop handler can be defined by the user to test for ranges by checking the definition of `\glxtrlocrangefmt`.

`displaystartloc` Start of a location range.

```

3549 \newcommand*{\glxtrdisplaystartloc}[2]{%
3550 \edef\glxtrlocrangefmt{#1}%
3551 \ifx\glxtrlocrangefmt\empty
3552 \def\glxtrlocrangefmt{glsnumberformat}%
3553 \fi
3554 \expandafter\glxtrdisplaysingleloc
3555 \expandafter{\glxtrlocrangefmt}{#2}%
3556 }

```


trdisplayendloc End of a location range.

```
3557 \newcommand*{\glxtrdisplayendloc}[2]{%
3558   \edef\@glxtr@tmp{#1}%
3559   \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{glsnumberformat}}{}%
3560   \ifx\glxtrlocrangefmt\@glxtr@tmp
3561   \else
3562     \GlossariesExtraWarning{Mismatched end location range
3563       (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
3564   \fi
3565   \expandafter\glxtrdisplayendloohook\expandafter{\@glxtr@tmp}{#2}%
3566   \expandafter\glxtrdisplaysingleloc
3567   \expandafter{\glxtrlocrangefmt}{#2}%
3568   \def\glxtrlocrangefmt{}%
3569 }
```

splayendloohook Allow the user to hook into the end of range command.

```
3570 \newcommand*{\glxtrdisplayendloohook}[2]{}
```

sxtrlocrangefmt Current range format. Empty if not in a range.

```
3571 \newcommand*{\glxtrlocrangefmt}{}%
```

ls@removespaces Redefine to allow adjustments to location hyperlink.

```
3572 \def\@gls@removespaces#1 #2\@nil{%
3573   \toks@=\expandafter{\the\toks@#1}%
3574   \ifx\#2\%
3575     \edef\x{\the\toks@}%
3576     \ifx\x\empty
3577     \else
3578       \glxtrlocationhyperlink{\gl Sentrycounter}{\@glo@counterprefix}{\the\toks@}%
3579     \fi
3580   \else
3581     \@gls@ReturnAfterFi{%
3582       \@gls@removespaces#2\@nil
3583     }%
3584   \fi
3585 }
```

cationhyperlink

```
3586 \newcommand*{\glxtrlocationhyperlink}[3]{%
3587   \ifdefvoid\glxtrsupplocationurl
3588   {%
3589     \hyperlink{#1#2#3}{#3}%
3590   }%
3591   {%
3592     \hyperref{\glxtrsupplocationurl}{#1#2#3}{#3}%
3593   }%
3594 }
```

supphypernumber

```
3595 \newcommand*{\glxtrsupphypernumber}[1]{%
3596 {%
3597   \glshasattribute{\glscurrententrylabel}{externallocation}%
3598   {%
3599     \def\glxtrsupplocationurl{%
3600       \glsggetattribute{\glscurrententrylabel}{externallocation}}%
3601   }%
3602   {%
3603     \def\glxtrsupplocationurl{}%
3604   }%
3605   \glshypernumber{#1}%
3606 }%
3607 }
```

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

@print@glossary

```
3608 \renewcommand{\@print@glossary}{%
3609   \makeatletter
3610   \@input@{\jobname.\csname @glo@type @in\endcsname}%
3611   \IfFileExists{\jobname.\csname @glo@type @in\endcsname}%
3612   {%
3613     {\glxtrNoGlossaryWarning{\@glo@type}}%
3614     \ifglxindy
3615       \ifcsundef{\xdy@\@glo@type @language}%
3616       {%
3617         \edef\@do@auxoutstuff{%
3618           \noexpand\AtEndDocument{%
3619             \noexpand\immediate\noexpand\write\@auxout{%
3620               \string\providecommand\string\@xdylanguage[2]{}}%
3621             \noexpand\immediate\noexpand\write\@auxout{%
3622               \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
3623           }%
3624         }%
3625       }%
3626     {%
3627       \edef\@do@auxoutstuff{%
3628         \noexpand\AtEndDocument{%
3629           \noexpand\immediate\noexpand\write\@auxout{%
3630             \string\providecommand\string\@xdylanguage[2]{}}%
3631           \noexpand\immediate\noexpand\write\@auxout{%
3632             \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
3633               @language\endcsname}}%
3634         }%
3635       }%
3636     }%
3637     \@do@auxoutstuff
```

```

3638 \edef\@do@auxoutstuff{%
3639 \noexpand\AtEndDocument{%
3640 \noexpand\immediate\noexpand\write\@auxout{%
3641 \string\providecommand\string\@gls@codepage[2]{}}%
3642 \noexpand\immediate\noexpand\write\@auxout{%
3643 \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
3644 }%
3645 }%
3646 \@do@auxoutstuff
3647 \fi
3648 \renewcommand*{\@warn@nomakeglossaries}{%
3649 \GlossariesWarningNoLine{\string\makeglossaries\space
3650 hasn't been used,^^Jthe glossaries will not be updated}%
3651 }%
3652 }

```

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

`\GlsWarningHead` Header message.

```

3653 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
3654 This document is incomplete. The external file associated with
3655 the glossary '#1' (which should be called \texttt{#2})
3656 hasn't been created.%
3657 }

```

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

```

3658 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
3659 This has probably happened because there are no entries defined
3660 in this glossary.%
3661 }

```

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

```

3662 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
3663 If you don't want this glossary,
3664 add \texttt{nomain} to your package option list when you load
3665 \texttt{glossaries-extra.sty}. For example:%
3666 }

```

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

```

3667 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
3668 Did you forget to use \texttt{type=#1} when you defined your
3669 entries? If you tried to load entries into this glossary with
3670 \texttt{\string\loadglsentries} did you remember to use
3671 \texttt{[#1]} as the optional argument? If you did, check that
3672 the definitions in the file you loaded all had the type set
3673 to \texttt{\string\glsdefaulttype}.%
3674 }

```

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

```

3675 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
3676   Check the contents of the file \texttt{#1}. If
3677   it's empty, that means you haven't indexed any of your entries in this
3678   glossary (using commands like \texttt{\string\gls} or
3679   \texttt{\string\glsadd}) so this list can't be generated.
3680   If the file isn't empty, the document build process hasn't been
3681   completed.%
3682 }

```

WarningAutoMake Message when automake option has been used.

```

3683 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
3684   You may need to rerun \LaTeX. If you already have, it may be that
3685   \TeX's shell escape doesn't allow you to run
3686   \ifglxindy xindy\else makeindex\fi. Check the
3687   transcript file \texttt{\jobname.log}. If the shell escape is
3688   disabled, try one of the following:
3689
3690   \begin{itemize}
3691     \item Run the external (Lua) application:
3692
3693         \texttt{makeglossaries-lite.lua \string\jobname\string}
3694
3695     \item Run the external (Perl) application:
3696
3697         \texttt{makeglossaries \string\jobname\string}
3698   \end{itemize}
3699
3700   Then rerun \LaTeX\ on this document.
3701   \GlossariesExtraWarning{Rerun required to build the
3702   glossary '#1' or check TeX's shell escape allows
3703   you to run \ifglxindy xindy\else makeindex\fi}%
3704 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

3705 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
3706   You need to either replace \texttt{\string\makenoidxglossaries}
3707   with \texttt{\string\makeglossaries} or replace
3708   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
3709   \texttt{\string\printnoidxglossary}
3710   (or \texttt{\string\printnoidxglossaries}) and then rebuild
3711   this document.%
3712 }

```

WarningBuildInfo Build advice.

```

3713 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
3714   Try one of the following:
3715   \begin{itemize}
3716     \item Add \texttt{automake} to your package option list when you load

```

```

3717         \texttt{glossaries-extra.sty}. For example:
3718
3719         \texttt{\string\usepackage[automake]%
3720             \glspopenbrace glossaries-extra\glsclosebrace}
3721
3722     \item Run the external (Lua) application:
3723
3724         \texttt{makeglossaries-lite.lua \string"\jobname\string"}
3725
3726     \item Run the external (Perl) application:
3727
3728         \texttt{makeglossaries \string"\jobname\string"}
3729 \end{itemize}
3730
3731 Then rerun \LaTeX\ on this document.%
3732 }

```

`oGlsWarningTail` Final paragraph.

```

3733 \newcommand{\GlsXtrNoGlsWarningTail}{%
3734 This message will be removed once the problem has been fixed.%
3735 }

```

`GlsWarningNoOut` No out file created. Build advice.

```

3736 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
3737 The file \texttt{#1} doesn't exist. This most likely means you haven't used
3738 \texttt{\string\makeglossaries} or you have used
3739 \texttt{\string\nofiles}. If this is just a draft version of the
3740 document, you can suppress this message using the
3741 \texttt{\nomissingglstext} package option.%
3742 }

```

`glossarywarning`

```

3743 \newcommand*{@@glstxr@defaultnoglossarywarning}[1]{%
3744 \glossarysection[\glossarytoctitle]{\glossarytitle}
3745 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@{@glo@type @in\endcsname}
3746 \par
3747 \glstxrifemptyglossary{#1}%
3748 {%
3749 \GlsXtrNoGlsWarningEmptyStart\space
3750 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
3751 \medskip
3752 \noindent\texttt{\string\usepackage[nomain\ifglstacronym ,acronym\fi]%
3753 \glspopenbrace glossaries-extra\glsclosebrace}
3754 \medskip
3755 }%
3756 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}}%
3757 }%
3758 {%
3759 \IfFileExists{\jobname.\csname @glotype@{@glo@type @out\endcsname}

```

```

3760 {%
3761   \GlsXtrNoGlsWarningCheckFile
3762     {\jobname.\csname @glotype@\@glo@type @out\endcsname}
3763
3764   \ifglautomake
3765
3766     \GlsXtrNoGlsWarningAutoMake{#1}
3767
3768   \else
3769
3770     \ifthenelse{\equal{#1}{main}}{%
3771       {%
3772         \GlsXtrNoGlsWarningEmptyMain\par
3773         \medskip
3774         \noindent\texttt{\string\usepackage[nomain]%
3775           \glsopenbrace glossaries-extra\glsclosebrace}
3776         \medskip
3777       }%
3778     }%
3779
3780     \ifdefequal\makeglossaries\@no@makeglossaries
3781     {%
3782       \GlsXtrNoGlsWarningMisMatch
3783     }%
3784     {%
3785       \GlsXtrNoGlsWarningBuildInfo
3786     }%
3787   \fi
3788 }%
3789 {%
3790   \GlsXtrNoGlsWarningNoOut
3791     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
3792   }%
3793 }%
3794 \par
3795 \GlsXtrNoGlsWarningTail
3796 }

```

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

xttrresourcefile Since it's dangerous for an external application to create a file with a .tex extension, as from v1.11 this enforces a .glstex extension to avoid conflict.

```

3797 \newcommand*{\glxtrresourcefile}[2] [] {%
3798   \glxtr@writefields
3799   \protected@write\@auxout{}\string\glxtr@resource{#1}{#2}}%
3800   \let\@glxtr@org@see@noindex\@gls@see@noindex
3801   \let\@gls@see@noindex\relax
3802   \IfFileExists{#2.glstex}%
3803   {%

```

Can't scope \@input so save and restore the category code of @ to allow for internal commands in the location list.

```

3804 \edef\@bibgls@restoreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
3805 \makeatletter
3806 \@input{#2.glstex}%
3807 \@bibgls@restoreat
3808 }%
3809 {%
3810 \GlossariesExtraWarning{No file '#2.glstex'}%
3811 }%
3812 \let\@gls@see@noindex\@glsxtr@org@see@noindex
3813 }
3814 \@onlypreamble\glsxtrresourcefile

```

trresourcecount

```

3815 \newcount\glsxtrresourcecount

```

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

```

3816 \newcommand*{\GlsXtrLoadResources}[1][{}]{%
3817 \ifnum\glsxtrresourcecount=0\relax
3818 \glsxtrresourcefile[#1]{\jobname}%
3819 \else
3820 \glsxtrresourcefile[#1]{\jobname-\the\glsxtrresourcecount}%
3821 \fi
3822 \advance\glsxtrresourcecount by 1\relax
3823 }

```

glsxtr@resource

```

3824 \newcommand*{\glsxtr@resource}[2]{}

```

\glsxtr@fields

```

3825 \newcommand*{\glsxtr@fields}[1]{}

```

xtr@texencoding

```

3826 \newcommand*{\glsxtr@texencoding}[1]{}

```

\glsxtr@langtag

```

3827 \newcommand*{\glsxtr@langtag}[1]{}

```

@pluralsuffixes

```

3828 \newcommand*{\glsxtr@pluralsuffixes}[4]{}

```

tr@shortcutsval

```

3829 \newcommand*{\glsxtr@shortcutsval}[1]{}

```

sxtr@linkprefix

```

3830 \newcommand*{\glsxtr@linkprefix}[1]{}

```

xttr@writefields This information only needs to be written once, so disable it after it's been used.

```

3831 \newcommand*{\glxtr@writefields}{%
3832   \protected@write\@auxout{}%
3833     {\string\providecommand*{\string\glxtr@fields}[1]{}}%
3834   \protected@write\@auxout{}%
3835     {\string\providecommand*{\string\glxtr@resource}[2]{}}%
3836   \protected@write\@auxout{}%
3837     {\string\providecommand*{\string\glxtr@pluralsuffixes}[4]{}}%
3838   \protected@write\@auxout{}%
3839     {\string\providecommand*{\string\glxtr@shortcutsval}[1]{}}%
3840   \protected@write\@auxout{}%
3841     {\string\providecommand*{\string\glxtr@linkprefix}[1]{}}%
3842   \protected@write\@auxout{}{\string\glxtr@fields{\@gls@keymap}}%

```

If any languages have been loaded, the language tag will be available in `\CurrentTrackedLanguageTag` (provided by `tracklang`). For multilingual documents, the required locale will have to be indicated in the sort key when using `\glxtrresourcefile`.

```

3843 \ifdef\CurrentTrackedLanguageTag
3844   {%
3845     \protected@write\@auxout{}{%
3846       \string\glxtr@langtag{\CurrentTrackedLanguageTag}}%
3847   }%
3848   {%
3849     \protected@write\@auxout{}{\string\glxtr@pluralsuffixes
3850       {\glsppluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
3851       {\glxtrabbrvpluralsuffix}}%
3852   \ifdef\inputencodingname
3853     {%
3854       \protected@write\@auxout{}{\string\glxtr@texencoding{\inputencodingname}}%
3855     }%
3856   {%

```

If `fontspec` has been loaded, assume UTF-8. (The encoding can be changed with `\XeTeXinputencoding`, but I can't work out how to determine the current encoding.)

```

3857     \@ifpackageloaded{fontspec}%
3858     {\protected@write\@auxout{}{\string\glxtr@texencoding{utf8}}}%
3859     {}%
3860   }%
3861   \protected@write\@auxout{}{\string\glxtr@shortcutsval{\@glxtr@shortcutsval}}%

```

Prefix deferred until the beginning of the document in case it's redefined later in the preamble. This is picked up by `bib2gls` when the external option is used.

```

3862 \AtBeginDocument
3863   {\protected@write\@auxout{}{\string\glxtr@linkprefix{\glolinkprefix}}}%
3864 \let\glxtr@writefields\relax

```

If the `automake` option is on, try running `bib2gls` if the aux file exists.

```

3865 \ifglautomake
3866   \IfFileExists{\jobname.aux}%
3867   {\immediate\write18{bib2gls "\jobname"}}{}%

```


If `\makeglossaries` is also used, allow `makeindex/xindy` to also be run, otherwise disable the error message about requiring `\makeglossaries` with `automake=true`.

```
3868 \ifx\@gls@doautomake\@gls@doautomake@err
3869 \let\@gls@doautomake\relax
3870 \fi
3871 \fi
3872 }
```

`do@automake@err`

```
3873 \newcommand*{\@gls@doautomake@err}{%
3874 \PackageError{glossaries}{You must use
3875 \string\makeglossaries\space with automake=true}
3876 {%
3877 Either remove the automake=true setting or
3878 add \string\makeglossaries\space to your document preamble.%
3879 }%
3880 }
```

Allow locations specific to a particular counter to be recorded.

`\glsxtr@record`

```
3881 \newcommand*{\glsxtr@record}[5]{}
```

`r@counterrecord` Aux file command.

```
3882 \newcommand*{\glsxtr@counterrecord}[3]{%
3883 \glsxtrfieldlistgadd{#1}{record.#2}{#3}%
3884 }
```

`unterrecordhook` Hook used by `\@glsxtr@dorecord`.

```
3885 \newcommand*{\@glsxtr@counterrecordhook}{}%
```

`trRecordCounter` Activate recording for a particular counter (identified in the argument).

```
3886 \newcommand*{\GlsXtrRecordCounter}[1]{%
3887 \@glsxtr@recordcounter{#1}%
3888 }
3889 \@onlypreamble\GlsXtrRecordCounter
```

`docounterrecord`

```
3890 \newcommand*{\@glsxtr@docounterrecord}[1]{%
3891 \protected@write\@auxout{}{\string\glsxtr@counterrecord
3892 {\@gls@label}{#1}{\csuse{the#1}}}%
3893 }
```

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

```
3894 \newcommand*{\printunsrtglossary}{%
3895 \@ifstar\s@printunsrtglossary\@printunsrtglossary
3896 }
```

ntunsrtglossary Unstarred version.

```
3897 \newcommand*{\@printunsrtglossary}[1][]{%
3898   \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
3899 }
```

ntunsrtglossary Starred version.

```
3900 \newcommand*{\s@printunsrtglossary}[2][]{%
3901   \begin{group}
3902     #2%
3903     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
3904   \end{group}
3905 }
```

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

```
3906 \newcommand*{\printunsrtglossaries}{%
3907   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
3908 }
```

@unsrt@glossary

```
3909 \newcommand*{\@print@unsrt@glossary}{%
3910   \glossarysection[\glossarytoctitle]{\glossarytitle}%
3911   \glossarypreamble
3912   check for empty list
3913   \glsxtrifemptyglossary{\@glo@type}%
3914   {%
3915     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
3916   }%
3917   \key@ifundefined{glossentry}{group}%
3918   {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
3919   {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
3920   \def\@gls@currentlettergroup{}}%
```

A loop within the tabular-like styles can cause problems, so move the loop outside.

```
3921 \def\@glsxtr@doglossary{%
3922   \begin{theglossary}%
3923   \glossaryheader
3924   \glsresetentrylist
3925 }%
3926 \expandafter\@for\expandafter\glscurrententrylabel\expandafter
3927   :\expandafter=\csname glolist@\@glo@type\endcsname\do{%
3928   \ifdefempty{\glscurrententrylabel}
3929   }%
3930   {\eappto\@glsxtr@doglossary{%
3931     \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}}%
3932 }
```

```

3933 \appto\@glstr@doglossary{\end{theglossary}}%
3934 \glstr@doglossary
3935 }%
3936 \glossarypostamble
3937 }

```

lossary@handler

```

3938 \newcommand{\@printunsrt@glossary@handler}[1]{%
3939 \xdef\glscurrententrylabel{#1}%
3940 \printunsrtglossaryhandler\glscurrententrylabel
3941 }

```

glossaryhandler

```

3942 \newcommand{\printunsrtglossaryhandler}[1]{%
3943 \glstrunsrtdo{#1}%
3944 }

```

srtglossaryunit

```

3945 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
3946 \s@printunsrtglossary[type=\gldefaulttype,#1]{%
3947 \printunsrtglossaryunitsetup{#2}%
3948 }%
3949 }

```

ossaryunitsetup

```

3950 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
3951 \renewcommand{\printunsrtglossaryhandler}[1]{%
3952 \glstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
3953 {\glstrunsrtdo{##1}}%
3954 }%
3955 }%
3956 \ifcsundef{theH#1}%
3957 {%
3958 \renewcommand*{\glolinkprefix}{record.#1.\csuse{the#1}.}%
3959 }%
3960 {%
3961 \renewcommand*{\glolinkprefix}{record.#1.\csuse{theH#1}.}%
3962 }%
3963 \renewcommand*{\glossarysection}[2][ ]{%
3964 \appto\glossarypostamble{\glspare\medskip\glspare}%
3965 }

```

srtglossaryunit

```

3966 \newcommand{\print@noop@unsrtglossaryunit}[2][ ]{%
3967 \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
3968 requires the record=only or record=alsoindex package option}{}%
3969 }

```

t@getgrouptitle

```
3970 \newrobustcmd*{\@glsxtr@unsrt@getgrouptitle}[2]{%
3971   \protected@edef\@glsxtr@titlelabel{glsxtr@grouptitle@#1}%
3972   \@onelevel@sanitize\@glsxtr@titlelabel
3973   \ifcsdef{\@glsxtr@titlelabel}
3974     {\letcs{#2}{\@glsxtr@titlelabel}}%
3975     {\def#2{#1}}%
3976 }
```

\glsxtrunsrtdo Provide a user-level call to \@glsxtr@noidx@do to make it easier to define a new handler.

```
3977 \newcommand{\glsxtrunsrtdo}{\@glsxtr@noidx@do}
```

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

```
3978 \newcommand{\@glsxtr@noidx@do}[1]{%
3979   \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3980   \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@location}%
3981   \ifglshasparent{#1}%
3982   {%
3983     \gls@level=\csuse{glo@\glsdetoklabel{#1}@level}\relax
3984     \ifdefvoid{\@gls@location}%
3985     {%
3986       \ifdefvoid{\@gls@loclist}%
3987       {%
3988         \subglossentry{\@gls@level}{#1}{}%
3989       }%
3990     }%
3991     \subglossentry{\@gls@level}{#1}%
3992     {%
3993       \glossaryentrynumbers{\@glsnoidxloclist{\@gls@loclist}}%
3994     }%
3995   }%
3996 }%
3997 {%
3998   \subglossentry{\@gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
3999 }%
4000 }%
4001 {%
4002   \key@ifundefined{glossentry}{group}%
4003   {%
4004     \letcs{\@gls@sort}{glo@\glsdetoklabel{#1}@sort}%
4005     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
4006   }%
4007   {%
4008     \protected@xdef\@glo@thislettergrp{%
4009       \csuse{glo@\glsdetoklabel{#1}@group}}%
4010   }%
4011   \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4012   {}%
```

```

4013   {%
4014     \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
4015     \expandafter\gls@groupheading\expandafter
4016       {\@glo@thislettergrp}%
4017   }%
4018   \global\let\@gls@currentlettergroup\@glo@thislettergrp
4019   \ifdefvoid{\@gls@location}%
4020   {%
4021     \ifdefvoid{\@gls@loclist}
4022     {%
4023       \glossentry{#1}{}%
4024     }%
4025     {%
4026       \glossentry{#1}%
4027     {%
4028       \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4029     }%
4030   }%
4031   }%
4032   {%
4033     \glossentry{#1}%
4034   {%
4035     \glossaryentrynumbers{\@gls@location}%
4036   }%
4037   }%
4038   }%
4039 }

```

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.

```

4040 \ifpackageloaded{glossaries-accsupp}
4041 {

```

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

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

```

4042   \newcommand*{\glsaccessname}[1]{%
4043     \glsnameaccessdisplay
4044     {%
4045       \glsentryname{#1}%
4046     }%
4047     {#1}%
4048   }

```

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

```

4049 \newcommand*{\Glsaccessname}[1]{%
4050   \glsnameaccessdisplay
4051   {%
4052     \Glsentryname{#1}%
4053   }%
4054   {#1}%
4055 }
```

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

```

4056 \newcommand*{\GLSaccessname}[1]{%
4057   \glsnameaccessdisplay
4058   {%
4059     \mfirstucMakeUppercase{\glsentryname{#1}}%
4060   }%
4061   {#1}%
4062 }
```

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

```

4063 \newcommand*{\glsaccesstext}[1]{%
4064   \glstextaccessdisplay
4065   {%
4066     \glsentrytext{#1}%
4067   }%
4068   {#1}%
4069 }
```

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

```

4070 \newcommand*{\Glsaccesstext}[1]{%
4071   \glstextaccessdisplay
4072   {%
4073     \Glsentrytext{#1}%
4074   }%
4075   {#1}%
4076 }
```

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

```

4077 \newcommand*{\GLSaccesstext}[1]{%
4078   \glstextaccessdisplay
4079   {%
4080     \mfirstucMakeUppercase{\glsentrytext{#1}}%
4081   }%
4082   {#1}%
4083 }
```

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

```

4084 \newcommand*{\glsaccessplural}[1]{%
4085   \glspluralaccessdisplay
4086   {%
4087     \glsentryplural{#1}%
4088   }%
4089   {#1}%
4090 }

```

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

```

4091 \newcommand*{\Glsaccessplural}[1]{%
4092   \glspluralaccessdisplay
4093   {%
4094     \Glsentryplural{#1}%
4095   }%
4096   {#1}%
4097 }

```

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

```

4098 \newcommand*{\GLSaccessplural}[1]{%
4099   \glspluralaccessdisplay
4100   {%
4101     \mfirstucMakeUppercase{\glsentryplural{#1}}%
4102   }%
4103   {#1}%
4104 }

```

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

```

4105 \newcommand*{\glsaccessfirst}[1]{%
4106   \glsfirstaccessdisplay
4107   {%
4108     \glsentryfirst{#1}%
4109   }%
4110   {#1}%
4111 }

```

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

```

4112 \newcommand*{\Glsaccessfirst}[1]{%
4113   \glsfirstaccessdisplay
4114   {%
4115     \Glsentryfirst{#1}%
4116   }%
4117   {#1}%
4118 }

```

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

```

4119 \newcommand*{\GLSaccessfirst}[1]{%

```

```

4120 \glsfirstaccessdisplay
4121 {%
4122     \mfirstucMakeUppercase{\glstentryfirst{#1}}%
4123 }%
4124 {#1}%
4125 }

```

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

```

4126 \newcommand*{\glsaccessfirstplural}[1]{%
4127     \glsfirstpluralaccessdisplay
4128     {%
4129         \glstentryfirstplural{#1}%
4130     }%
4131     {#1}%
4132 }

```

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

```

4133 \newcommand*{\Glsaccessfirstplural}[1]{%
4134     \glsfirstpluralaccessdisplay
4135     {%
4136         \Glstentryfirstplural{#1}%
4137     }%
4138     {#1}%
4139 }

```

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

```

4140 \newcommand*{\GLSaccessfirstplural}[1]{%
4141     \glsfirstpluralaccessdisplay
4142     {%
4143         \mfirstucMakeUppercase{\glstentryfirstplural{#1}}%
4144     }%
4145     {#1}%
4146 }

```

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

```

4147 \newcommand*{\glsaccesssymbol}[1]{%
4148     \glssymbolaccessdisplay
4149     {%
4150         \glstentrysymbol{#1}%
4151     }%
4152     {#1}%
4153 }

```

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

```

4154 \newcommand*{\Glsaccesssymbol}[1]{%
4155     \glssymbolaccessdisplay

```



```

4156     {%
4157         \Glsentrysymbol{#1}%
4158     }%
4159     {#1}%
4160 }

```

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

```

4161 \newcommand*{\GLSaccesssymbol}[1]{%
4162     \glssymbolaccessdisplay
4163     {%
4164         \mfirstucMakeUppercase{\Glsentrysymbol{#1}}%
4165     }%
4166     {#1}%
4167 }

```

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

```

4168 \newcommand*{\glssaccesssymbolplural}[1]{%
4169     \glssymbolpluralaccessdisplay
4170     {%
4171         \Glsentrysymbolplural{#1}%
4172     }%
4173     {#1}%
4174 }

```

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

```

4175 \newcommand*{\Glsaccesssymbolplural}[1]{%
4176     \glssymbolpluralaccessdisplay
4177     {%
4178         \Glsentrysymbolplural{#1}%
4179     }%
4180     {#1}%
4181 }

```

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

```

4182 \newcommand*{\GLSaccesssymbolplural}[1]{%
4183     \glssymbolpluralaccessdisplay
4184     {%
4185         \mfirstucMakeUppercase{\Glsentrysymbolplural{#1}}%
4186     }%
4187     {#1}%
4188 }

```

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

```

4189 \newcommand*{\glsaccessdesc}[1]{%
4190     \glsdescriptionaccessdisplay
4191     {%
4192         \Glsentrydesc{#1}%

```

```

4193     }%
4194     {#1}%
4195 }

```

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

```

4196 \newcommand*{\Glsaccessdesc}[1]{%
4197   \glsdescriptionaccessdisplay
4198   {%
4199     \Glsentrydesc{#1}%
4200   }%
4201   {#1}%
4202 }

```

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

```

4203 \newcommand*{\GLSaccessdesc}[1]{%
4204   \glsdescriptionaccessdisplay
4205   {%
4206     \mfirstucMakeUppercase{\glsentrydesc{#1}}%
4207   }%
4208   {#1}%
4209 }

```

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

```

4210 \newcommand*{\glsaccessdescplural}[1]{%
4211   \glsdescriptionpluralaccessdisplay
4212   {%
4213     \glsentrydescplural{#1}%
4214   }%
4215   {#1}%
4216 }

```

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

```

4217 \newcommand*{\Glsaccessdescplural}[1]{%
4218   \glsdescriptionpluralaccessdisplay
4219   {%
4220     \Glsentrydescplural{#1}%
4221   }%
4222   {#1}%
4223 }

```

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

```

4224 \newcommand*{\GLSaccessdescplural}[1]{%
4225   \glsdescriptionpluralaccessdisplay
4226   {%
4227     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
4228   }%

```

```

4229     {#1}%
4230 }

```

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

```

4231 \newcommand*\glsaccessshort}[1]{%
4232   \glsshortaccessdisplay
4233   {%
4234     \glentryshort{#1}%
4235   }%
4236   {#1}%
4237 }

```

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

```

4238 \newcommand*\Glsaccessshort}[1]{%
4239   \glsshortaccessdisplay
4240   {%
4241     \Glsentryshort{#1}%
4242   }%
4243   {#1}%
4244 }

```

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

```

4245 \newcommand*\GLSaccessshort}[1]{%
4246   \glsshortaccessdisplay
4247   {%
4248     \mfirstucMakeUppercase{\glentryshort{#1}}%
4249   }%
4250   {#1}%
4251 }

```

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

```

4252 \newcommand*\lsaccessshortpl}[1]{%
4253   \glsshortpluralaccessdisplay
4254   {%
4255     \glentryshortpl{#1}%
4256   }%
4257   {#1}%
4258 }

```

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

```

4259 \newcommand*\Lsaccessshortpl}[1]{%
4260   \glsshortpluralaccessdisplay
4261   {%
4262     \Lsentryshortpl{#1}%
4263   }%
4264   {#1}%
4265 }

```

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

```
4266 \newcommand*{\LSaccessshortpl}[1]{%
4267   \glsshortpluralaccessdisplay
4268   {%
4269     \mfirstucMakeUppercase{\glsentryshortpl{#1}}%
4270   }%
4271   {#1}%
4272 }
```

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

```
4273 \newcommand*{\glsaccesslong}[1]{%
4274   \glslongaccessdisplay{\glsentrylong{#1}}{#1}%
4275 }
```

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

```
4276
4277 \newcommand*{\Glsaccesslong}[1]{%
4278   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
4279 }
```

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

```
4280 \newcommand*{\GLSaccesslong}[1]{%
4281   \glslongaccessdisplay
4282   {%
4283     \mfirstucMakeUppercase{\glsentrylong{#1}}%
4284   }%
4285   {#1}%
4286 }
```

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

```
4287 \newcommand*{\glsaccesslongpl}[1]{%
4288   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
4289 }
```

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

```
4290
4291 \newcommand*{\Glsaccesslongpl}[1]{%
4292   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
4293 }
```

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

```
4294 \newcommand*{\GLSaccesslongpl}[1]{%
4295   \glslongpluralaccessdisplay
4296   {%
4297     \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
4298   }%
4299   {#1}%
4300 }
```

End of if part

4301 }

4302 {

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

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

4303 `\newcommand*{\glsaccessname}[1]{\glentryname{#1}}`

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

4304 `\newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}`

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

4305 `\newcommand*{\GLSaccessname}[1]{%`

4306 `\protect\mfirstucMakeUppercase{\glentryname{#1}}}`

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

4307 `\newcommand*{\glsaccesstext}[1]{\glentrytext{#1}}`

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

4308 `\newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}`

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

4309 `\newcommand*{\GLSaccesstext}[1]{%`

4310 `\protect\mfirstucMakeUppercase{\glentrytext{#1}}}`

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

4311 `\newcommand*{\glsaccessplural}[1]{\glentryplural{#1}}`

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

4312 `\newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}`

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

4313 `\newcommand*{\GLSaccessplural}[1]{%`

4314 `\protect\mfirstucMakeUppercase{\glentryplural{#1}}}`

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

4315 `\newcommand*{\glsaccessfirst}[1]{\glentryfirst{#1}}`

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

4316 `\newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}`

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

```

4317 \newcommand*{\GLSaccessfirst}[1]{%
4318 \protect\mfirstucMakeUppercase{\glentryfirst{#1}}}
```

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

```

4319 \newcommand*{\glsaccessfirstplural}[1]{\glentryfirstplural{#1}}
```

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

```

4320 \newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}
```

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

```

4321 \newcommand*{\GLSaccessfirstplural}[1]{%
4322 \protect\mfirstucMakeUppercase{\glentryfirstplural{#1}}}
```

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

```

4323 \newcommand*{\glsaccesssymbol}[1]{\glentrysymbol{#1}}
```

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

```

4324 \newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}
```

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

```

4325 \newcommand*{\GLSaccesssymbol}[1]{%
4326 \protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}
```

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

```

4327 \newcommand*{\glsaccesssymbolplural}[1]{\glentrysymbolplural{#1}}
```

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

```

4328 \newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}
```

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

```

4329 \newcommand*{\GLSaccesssymbolplural}[1]{%
4330 \protect\mfirstucMakeUppercase{\glentrysymbolplural{#1}}}
```

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

```

4331 \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.

```

4332 \newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}
```

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

```

4333 \newcommand*{\GLSaccessdesc}[1]{%
4334 \protect\mfirstucMakeUppercase{\glentrydesc{#1}}}
```

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

```

4335 \newcommand*{\glsaccessdescplural}[1]{\glentrydescplural{#1}}
```

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

```

4336 \newcommand*{\Glsaccessdescplural}[1]{\Glentrydescplural{#1}}
```

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

```

4337 \newcommand*{\GLSaccessdescplural}[1]{%
4338 \protect\mfirstucMakeUppercase{\glentrydescplural{#1}}}
```

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

```

4339 \newcommand*{\glsaccessshort}[1]{\glentryshort{#1}}
```

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

```

4340 \newcommand*{\Glsaccessshort}[1]{\Glentryshort{#1}}
```

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

```

4341 \newcommand*{\GLSaccessshort}[1]{%
4342 \protect\mfirstucMakeUppercase{\glentryshort{#1}}}
```

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

```

4343 \newcommand*{\glsaccessshortpl}[1]{\glentryshortpl{#1}}
```

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

```

4344 \newcommand*{\Glsaccessshortpl}[1]{\Glentryshortpl{#1}}
```

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

```

4345 \newcommand*{\GLSaccessshortpl}[1]{%
4346 \protect\mfirstucMakeUppercase{\glentryshortpl{#1}}}
```

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

```

4347 \newcommand*{\glsaccesslong}[1]{\glentrylong{#1}}
```

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

```

4348 \newcommand*{\Glsaccesslong}[1]{\Glentrylong{#1}}
```

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

```

4349 \newcommand*{\GLSaccesslong}[1]{%
4350 \protect\mfirstucMakeUppercase{\glentrylong{#1}}}
```

`glsaccesslongpl` Display the long plural form (no link and no check for existence).
 4351 `\newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}`

`Glsaccesslongpl` Display the long plural form (no link and no check for existence).
 4352 `\newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}`

`GLSaccesslongpl` Display the longplural value (no link and no check for existence). converted to upper case.
 4353 `\newcommand*{\GLSaccesslongpl}[1]{%`
 4354 `\protect\mfirstucMakeUppercase{\glsentrylongpl{#1}}}`
 End of else part
 4355 }

1.5 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.
 4356 `\glsaddstoragekey{category}{general}{\glscategory}`

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.
 4357 `\newcommand{\glsifcategory}[4]{%`
 4358 `\ifglsfieldeq{#1}{category}{#2}{#3}{#4}%`
 4359 }

Categories can have attributes.

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

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

4360 `\newcommand*{\glssetcategoryattribute}[3]{%`
 4361 `\csdef{@glsxtr@categoryattr@@#1@#2}{#3}%`
 4362 }

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

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

4363 `\newcommand*{\glsgetcategoryattribute}[2]{%`
 4364 `\csuse{@glsxtr@categoryattr@@#1@#2}%`
 4365 }

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

Tests if the category has the given attribute set.

```
4366 \newcommand*{\glshascategoryattribute}[4]{%
4367   \ifcsvoid{@glstr@categoryattr@#1@#2}{#4}{#3}%
4368 }
```

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

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

```
4369 \newcommand*{\glissetattribute}[3]{%
4370   \glsetcategoryattribute{\glscategory{#1}}{#2}{#3}%
4371 }
```

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

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

```
4372 \newcommand*{\glsggetattribute}[2]{%
4373   \glsgcategoryattribute{\glscategory{#1}}{#2}%
4374 }
```

\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.

```
4375 \newcommand*{\glshasattribute}[4]{%
4376   \ifglstryexists{#1}%
4377   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
4378   {#4}%
4379 }
```

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

True if category has the attribute with the given value.

```
4380 \newcommand{\glisifcategoryattribute}[5]{%
```

```

4381 \ifcsundef{@glxtr@categoryattr@@#1@#2}%
4382 {#5}%
4383 {\ifcsstring{@glxtr@categoryattr@@#1@#2}{#3}{#4}{#5}}%
4384 }

```

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

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

```

4385 \newcommand{\glsifattribute}[5]{%
4386   \ifglentryexists{#1}%
4387   {\glsifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
4388   {#5}%
4389 }

```

Set attributes for the default general category:

```
4390 \glsetcategoryattribute{general}{regular}{true}
```

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

```
4391 \glsetcategoryattribute{acronym}{regular}{true}
```

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

```

4392 \newcommand*{\glsetregularcategory}[1]{%
4393   \glsetcategoryattribute{#1}{regular}{true}%
4394 }

```

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

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

```

4395 \newcommand{\glsifregularcategory}[3]{%
4396   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
4397 }

```

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

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

```

4398 \newcommand{\glsifnotregularcategory}[3]{%
4399   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
4400 }

```

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

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

```
4401 \newcommand{\glsifregular}[3]{%
4402   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
4403 }
```

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

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

```
4404 \newcommand{\glsifnotregular}[3]{%
4405   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
4406 }
```

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

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

```
4407 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
4408   \forallglossaries[#1]{#3}%
4409   {%
4410     \forlentries[#3]{#4}%
4411     {%
4412       \glsifcategory{#4}{#2}{#5}{}%
4413     }%
4414   }%
4415 }
```

`\glsforeachwithattribute` `\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.

```

4416 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
4417   \forallglossaries[#1]{#4}%
4418   {%
4419     \forglsentries[#4]{#5}%
4420     {%
4421       \glsifattribute{#5}{#2}{#3}{#6}{}%
4422     }%
4423   }%
4424 }

```

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

```

4425 \ifdef\newterm
4426 {%

```

`\newterm`

```

4427   \renewcommand*{\newterm}[2][ ]{%
4428     \newglossaryentry{#2}%
4429     {type={index},category=index,name={#2},%
4430      description={\glstrpostdescription\nopostdesc},#1}%
4431   }

```

Indexed terms are regular by default.

```

4432   \glsssetcategoryattribute{index}{regular}{true}

```

`trpostdescindex`

```

4433   \newcommand*{\glstrpostdescindex}{}
4434 }
4435 {}

```

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`.

```

4436 \ifdef\printsymbols
4437 {%

```

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

```

4438   \newcommand*{\glstrnewsymbol}[3][ ]{%
4439     \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
4440   }

```

Symbols are regular by default.

```

4441   \glsssetcategoryattribute{symbol}{regular}{true}

```

`rpostdescsymbol`

```

4442   \newcommand*{\glstrpostdescsymbol}{}

```

```
4443 }
4444 {}
```

Similar for the numbers option.

```
4445 \ifdef\printnumbers
4446 {%
```

glsxtrnewnumber

```
4447 \ifdef\printnumbers
4448   \newcommand*{\glsxtrnewnumber}[3] [] {%
4449     \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
4450   }
```

Numbers are regular by default.

```
4451   \glssetcategoryattribute{number}{regular}{true}
```

rpostdescnumber

```
4452   \newcommand*{\glsxtrpostdescnumber}{}

4453 }
4454 {}
```

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

```
4455 \newcommand*{\glsxtrsetcategory}[2] {%
4456   \@for\@glsxtr@label:=#1\do
4457   {%
4458     \glsfieldxdef{\@glsxtr@label}{category}{#2}%
4459   }%
4460 }
```

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

```
4461 \newcommand*{\glsxtrsetcategoryforall}[2] {%
4462   \forallglossaries[#1]{\@glsxtr@type}{%
4463     \forglsentries[\@glsxtr@type]{\@glsxtr@label}%
4464     {%
4465       \glsfieldxdef{\@glsxtr@label}{category}{#2}%
4466     }%
4467   }%
4468 }
```

trfieldtitlecase

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

Apply title casing to the contents of the given field.

```
4469 \newcommand*{\glsxtrfieldtitlecase}[2] {%
```

```

4470 \expandafter\glxtrfieldtitlecasecs\expandafter
4471   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
4472 }

```

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

```

4473 \newcommand*{\glxtrfieldtitlecasecs}[1]{\xcapitalisewords{#1}}

```

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

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

```

4474 \ifpackageloaded{glossaries-accsupp}
4475 {
4476   \renewcommand*{\glossentrydesc}[1]{%
4477     \glsdoifexistsorwarn{#1}%
4478     {%
4479       \glssetabbrvfmt{\glscategory{#1}}%

```

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

```

4480   \glshasattribute{#1}{glossdescfont}%
4481   {%
4482     \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
4483     \ifcsdef{\@glxtr@attrval}%
4484     {%
4485       \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
4486     }%
4487     {%
4488       \GlossariesExtraWarning{Unknown control sequence name
4489         ‘\@glxtr@attrval’ supplied in glossdescfont attribute
4490         for entry ‘#1’. Ignoring}%
4491       \let\@glxtr@glossdescfont\@firstofone
4492     }%
4493   }%
4494   {\let\@glxtr@glossdescfont\@firstofone}%
4495   \glsifattribute{#1}{glossdesc}{firstuc}%
4496   {%
4497     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
4498   }%
4499   {%
4500     \glsifattribute{#1}{glossdesc}{title}%
4501     {%
4502       \@glxtr@do@titlecaps@warn
4503       \glsdescriptionaccessdisplay
4504       {%
4505         \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
4506       }%
4507       {#1}%

```

```

4508     }%
4509     {%
4510         \@glxtr@glossdescfont{\glsaccessdesc{#1}}%
4511     }%
4512 }%
4513 }%
4514 }
4515 }
4516 {
4517 \renewcommand*{\glossentrydesc}[1]{%
4518     \glsdoifexistsorwarn{#1}%
4519     {%
4520         \glssetabbrvfmt{\glscategory{#1}}%
4521         \glsattribute{#1}{glossdescfont}%
4522         {%
4523             \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
4524             \ifcsdef{\@glxtr@attrval}%
4525             {%
4526                 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
4527             }%
4528             {%
4529                 \GlossariesExtraWarning{Unknown control sequence name
4530                     '\@glxtr@attrval' supplied in glossdescfont attribute
4531                     for entry '#1'. Ignoring}%
4532                 \let\@glxtr@glossdescfont\@firstofone
4533             }%
4534         }%
4535         {\let\@glxtr@glossdescfont\@firstofone}%
4536         \glsattribute{#1}{glossdesc}{firstuc}%
4537         {%
4538             \@glxtr@glossdescfont{\Glsentrydesc{#1}}%
4539         }%
4540         {%
4541             \glsattribute{#1}{glossdesc}{title}%
4542             {%
4543                 \@glxtr@do@titlecaps@warn
4544                 \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
4545             }%
4546             {%
4547                 \@glxtr@glossdescfont{\glsentrydesc{#1}}%
4548             }%
4549         }%
4550     }%
4551 }
4552 }

```

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

```

4553 \ifpackageloaded{glossaries-accsupp}

```

```

4554 {
4555   \renewcommand*{\glossentryname}[1]{%
4556     \@glsdoifexistsorwarn{#1}%
4557     {%
4558       \glsetabbrvfmt{\glscategory{#1}}%

```

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

```

4559     \glshasattribute{#1}{glossnamefont}%
4560     {%
4561       \edef\@glsxtr@attrval{\glsetattribute{#1}{glossnamefont}}%
4562       \ifcsdef{\@glsxtr@attrval}%
4563       {%
4564         \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
4565       }%
4566       {%
4567         \GlossariesExtraWarning{Unknown control sequence name
4568           '\@glsxtr@attrval' supplied in glossnamefont attribute
4569           for entry '#1'. Reverting to default \string\glsnamefont}%
4570         \let\@glsxtr@glossnamefont\glsnamefont
4571       }%
4572     }%
4573     {\let\@glsxtr@glossnamefont\glsnamefont}%
4574     \glsifattribute{#1}{glossname}{firstuc}%
4575     {%
4576       \glsnameaccessdisplay
4577       {%
4578         \@glsxtr@glossnamefont{\Glsentryname{#1}}%
4579       }%
4580       {#1}%
4581     }%
4582     {%
4583       \glsifattribute{#1}{glossname}{title}%
4584       {%
4585         \@glsxtr@do@titlecaps@warn
4586         \glsnameaccessdisplay
4587         {%
4588           \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
4589         }%
4590         {#1}%
4591       }%
4592       {%
4593         \glsifattribute{#1}{glossname}{uc}%
4594         {%
4595           \glsnameaccessdisplay
4596           {%

```

Hide the label from the upper-casing command.

```

4597         \letcs{\glo@name}{\glo\glsdetoklabel{#1}@name}%
4598         \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
4599       }%

```



```

4600         {#1}%
4601     }%
4602     {%
4603         \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@name}%
4604         \glsnameaccessdisplay
4605         {%
4606             \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
4607             }%
4608         {#1}%
4609     }%
4610 }%
4611 }%

```

Do post-name hook:

```

4612     \glxtrpostnamehook{#1}%
4613 }%
4614 }
4615 }
4616 {
4617     \renewcommand*{\glossentryname}[1]{%
4618         \@glsdoifexistsorwarn{#1}%
4619         {%
4620             \glssetabbrvfmt{\glscategory{#1}}%
4621             \glsattribute{#1}{glossnamefont}%
4622             {%
4623                 \edef\@glxtr@attrval{\glsattribute{#1}{glossnamefont}}%
4624                 \ifcsdef{\@glxtr@attrval}%
4625                 {%
4626                     \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
4627                 }%
4628                 {%
4629                     \GlossariesExtraWarning{Unknown control sequence name
4630                     '\@glxtr@attrval' supplied in glossnamefont attribute
4631                     for entry '#1'. Reverting to default \string\glsnamefont}%
4632                     \let\@glxtr@glossnamefont\glsnamefont
4633                 }%
4634             }%
4635             {\let\@glxtr@glossnamefont\glsnamefont}%
4636             \glsifattribute{#1}{glossname}{firstuc}%
4637             {%
4638                 \@glxtr@glossnamefont{\Glsentryname{#1}}%
4639             }%
4640             {%
4641                 \glsifattribute{#1}{glossname}{title}%
4642                 {%
4643                     \@glxtr@do@titlecaps@warn
4644                     \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
4645                 }%
4646                 {%
4647                     \glsifattribute{#1}{glossname}{uc}%

```

```
4648         {%
```

Hide the label from the upper-casing command.

```
4649         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
4650         \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
4651     }%
4652     {%
```

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.

```
4653         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
4654         \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
4655     }%
4656 }%
4657 }
```

Do post-name hook.

```
4658     \glstrpostnamehook{#1}%
4659 }%
4660 }
4661 }
```

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

```
4662 \ifpackageloaded{glossaries-accsupp}
4663 {
4664     \renewcommand*{\Glossentryname}[1]{%
4665         \@glstdoifexistsorwarn{#1}%
4666         {%
4667             \glsetabbrvfmt{\glscategory{#1}}%
```

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

```
4668         \glshasattribute{#1}{glossnamefont}%
4669     {%
4670         \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
4671         \ifcsdef{\@glstr@attrval}%
4672     {%
4673         \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
4674     }%
4675     {%
4676         \GlossariesExtraWarning{Unknown control sequence name
4677             ‘\@glstr@attrval’ supplied in glossnamefont attribute
4678             for entry ‘#1’. Reverting to default \string\glsnamefont}%
4679         \let\@glstr@glossnamefont\glsnamefont
4680     }%
4681 }%
4682 {\let\@glstr@glossnamefont\glsnamefont}%
4683 \glsnameaccessdisplay
4684 {%
4685     \@glstr@glossnamefont{\Glsentryname{#1}}%
4686 }%
4687 {#1}%
```

Do post-name hook:

```
4688 \glxstrpostnamehook{#1}%
4689 }%
4690 }
4691 }
4692 {
4693 \renewcommand*{\Glossentryname}[1]{%
4694 \@glxstrdoifexistsorwarn{#1}%
4695 {%
4696 \glxstrsetabbrvfmt{\glxstrcategory{#1}}%
4697 \glxstrhasattribute{#1}{\glxstrglossnamefont}%
4698 {%
4699 \edef\@glxstr@attrval{\glxstrgetattribute{#1}{\glxstrglossnamefont}}%
4700 \ifcsdef{\@glxstr@attrval}%
4701 {%
4702 \letcs{\@glxstr@glossnamefont}{\@glxstr@attrval}%
4703 }%
4704 {%
4705 \GlossariesExtraWarning{Unknown control sequence name
4706 '\@glxstr@attrval' supplied in glossnamefont attribute
4707 for entry '#1'. Reverting to default \string\glxstrglossnamefont}%
4708 \let\@glxstr@glossnamefont\glxstrglossnamefont
4709 }%
4710 }%
4711 {\let\@glxstr@glossnamefont\glxstrglossnamefont}%
4712 \@glxstr@glossnamefont{\Glossentryname{#1}}%

```

Do post-name hook:

```
4713 \glxstrpostnamehook{#1}%
4714 }%
4715 }
4716 }
```

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.

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

```
4717 \newcommand*{\glxstrpostnamehook}[1]{%
4718 \def\@glxstrnumberformat{\glxstrnumberformat}%
4719 \glxstrdoautoindexname{#1}{\indexname}%

```

Allow categories to hook in here.

```
4720 \csuse{\glxstrpostname\glxstrcategory{#1}}%
4721 }
```

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

```
4722 \newif\if@glxstr@format@override
4723 \@glxstr@format@overridefalse

```

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

`xFormatOverride`

```

4724 \ifpackage{hyperref}
4725 {
    If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
    don't add it.
4726   \ifHy@hyperindex
4727     \newcommand*\GlsXtrEnableIndexFormatOverride{%
4728       \@glstr@format@overridetrue
4729       \appto\theindex{\let\glshypernumber\@firstofone}%
4730     }
4731   \else
4732     \newcommand*\GlsXtrEnableIndexFormatOverride{%
4733       \@glstr@format@overridetrue
4734       \appto\theindex{\let\glshypernumber\hyperpage}%
4735     }
4736   \fi
4737 }
4738 {
4739   \newcommand*\GlsXtrEnableIndexFormatOverride{%
4740     \@glstr@format@overridetrue
4741   }
4742 }
4743 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

`doautoindexname`

```

4744 \newcommand*\glstrdoautoindexname[2]{%
4745   \glshasattribute{#1}{#2}%
4746   {%
    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.
4747     \@glstr@autoindex@setname{#1}%
    If the attribute value is simply "true" don't add an encap, otherwise use the value as the encap.
4748     \protected@edef\@glstr@attrval{\glsggetattribute{#1}{#2}}%
4749     \if@glstr@format@override
4750       \ifdefstring{\@glstr@attrval}{\glstr@attrval}{%
4751         {\let\@glstr@attrval\@glstr@attrval}%
4752       }%
4753     \ifdefstring{\@glstr@attrval}{true}%
4754     {}%
4755     {\eappto\@glo@name{\@glstr@autoindex@encap\@glstr@attrval}}%
4756     \expandafter\glstrautoindex\expandafter{\@glo@name}%
4757   }%
4758   {}%
4759 }

```

glxtrautoindex

```
4760 \newcommand*{\glxtrautoindex}{\index}
```

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

```
4761 \newcommand*{\@glxtr@autoindex@setname}[1]{%
4762   \protected@edef\@glo@name{\glxtrautoindexentry{#1}}%
4763   \glxtrautoindexassignsort{\@glo@sort}{#1}%
4764   \@gls@checkmkidxchars\@glo@sort
4765   \@glxtr@autoindex@doextra@esc\@glo@sort
4766   \epreto\@glo@name{\@glo@sort\@glxtr@autoindex@at}%
4767 }
```

rautoindexentry Command used for the actual part when auto-indexing.

```
4768 \newcommand*{\glxtrautoindexentry}[1]{\string\glxentryname{#1}}
```

trautoindexsort Used to assign the sort value when auto-indexing.

```
4769 \newcommand*{\glxtrautoindexassignsort}[2]{%
4770   \glsletentryfield{#1}{#2}{sort}%
4771 }
```

dex@doextra@esc

```
4772 \newcommand*{\@glxtr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
4773   \ifx\@glxtr@autoindex@esc\@gls@quotechar
4774   \else
4775     \def\@gls@checkedmkidx{%
4776       \edef\@glxtr@checkspch{%
4777         \noexpand\@glxtr@autoindex@escquote\expandonce{#1}%
4778         \noexpand\@empty\@glxtr@autoindex@esc\noexpand\@nnil
4779         \@glxtr@autoindex@esc\noexpand\@empty\noexpand\@glxtr@endescspch}%
4780       \@glxtr@checkspch
4781       \let#1\@gls@checkedmkidx\relax
4782     \fi
```

Escape actual character unless it has already been escaped.

```
4783   \ifx\@glxtr@autoindex@at\@gls@actualchar
4784   \else
4785     \def\@gls@checkedmkidx{%
4786       \edef\@glxtr@checkspch{%
4787         \noexpand\@glxtr@autoindex@escat\expandonce{#1}%
4788         \noexpand\@empty\@glxtr@autoindex@at\noexpand\@nnil
4789         \@glxtr@autoindex@at\noexpand\@empty\noexpand\@glxtr@endescspch}%
4790       \@glxtr@checkspch
4791       \let#1\@gls@checkedmkidx\relax
4792     \fi
```

Escape level character unless it has already been escaped.

```
4793   \ifx\@glxtr@autoindex@level\@gls@levelchar
4794   \else
```

```

4795 \def\@gls@checkedmkidx{%
4796 \edef\@@glsxtr@checkspch{%
4797 \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
4798 \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
4799 \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4800 @@glsxtr@checkspch
4801 \let#1\@gls@checkedmkidx\relax
4802 \fi

```

Escape encap character unless it has already been escaped.

```

4803 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
4804 \else
4805 \def\@gls@checkedmkidx{%
4806 \edef\@@glsxtr@checkspch{%
4807 \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
4808 \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
4809 \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4810 @@glsxtr@checkspch
4811 \let#1\@gls@checkedmkidx\relax
4812 \fi
4813 }

```

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`.

```

4814 \newcommand*{\@glsxtr@autoindex@at}{}

```

`\trSetActualChar` Set the actual character.

```

4815 \newcommand*{\GlsXtrSetActualChar}[1]{%
4816 \gdef\@glsxtr@autoindex@at{#1}%
4817 \def\@glsxtr@autoindex@escat##1##2##3\@glsxtr@endescspch{%
4818 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escat}{##1}{##2}{##3}%
4819 }%
4820 }
4821 \@onlypreamble\GlsXtrSetActualChar
4822 \makeatother
4823 \GlsXtrSetActualChar{@}
4824 \makeatletter

```

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

```

4825 \newcommand*{\@glsxtr@autoindex@encap}{}

```

`\XtrSetEncapChar` Set the encap character.

```

4826 \newcommand*{\GlsXtrSetEncapChar}[1]{%
4827 \gdef\@glsxtr@autoindex@encap{#1}%
4828 \def\@glsxtr@autoindex@escencap##1##2##3\@glsxtr@endescspch{%
4829 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escencap}{##1}{##2}{##3}%
4830 }%

```

```

4831 }
4832 \GlsXtrSetEncapChar{||}
4833 \@onlypreamble\GlsXtrSetEncapChar

autoindex@level  Level character for use with \index.
4834 \newcommand*{\@glxtr@autoindex@level}{%

XtrSetLevelChar  Set the encap character.
4835 \newcommand*{\GlsXtrSetLevelChar}[1]{%
4836   \gdef\@glxtr@autoindex@level{#1}%
4837   \def\@glxtr@autoindex@esclevel##1#1##2#1##3\@glxtr@endescspch{%
4838     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@esclevel}{##1}{##2}{##3}%
4839   }%
4840 }
4841 \GlsXtrSetLevelChar{!}
4842 \@onlypreamble\GlsXtrSetLevelChar

r@autoindex@esc  Escape character for use with \index.
4843 \newcommand*{\@glxtr@autoindex@esc}{"}

lsXtrSetEscChar  Set the escape character.
4844 \newcommand*{\GlsXtrSetEscChar}[1]{%
4845   \gdef\@glxtr@autoindex@esc{#1}%
4846   \def\@glxtr@autoindex@escquote##1#1##2#1##3\@glxtr@endescspch{%
4847     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escquote}{##1}{##2}{##3}%
4848   }%
4849 }
4850 \GlsXtrSetEscChar{"}
4851 \@onlypreamble\GlsXtrSetEscChar

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:
4852 \ifdef\actualchar
4853 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
4854 {}

Quote character \quotechar:
4855 \ifdef\quotechar
4856 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
4857 {}

Level character \levelchar:
4858 \ifdef\levelchar
4859 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
4860 {}

Encap character \encapchar:
4861 \ifdef\encapchar
4862 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
4863 {}

```

leto@endescspch

```
4864 \def\@glsxtr@gobbleto@endescspch#1\@glsxtr@endescspch{}
```

toindex@esc@spch

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

```
4865 \newcommand*{\@glsxtr@autoindex@escspch}[5]{%
4866   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
4867   \toks@={#3}%
4868   \ifx\@nnil#3\relax
4869     \def\@glsxtr@checkspch{\@glsxtr@gobbleto@endescspch#5\@glsxtr@endescspch}%
4870   \else
4871     \ifx\@nnil#4\relax
4872       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
4873       \def\@glsxtr@checkspch{\@glsxtr@gobbleto@endescspch
4874         #4#5\@glsxtr@endescspch}%
4875     \else
4876       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
4877         \@glsxtr@autoindex@esc#1}%
4878       \def\@glsxtr@checkspch{#2#5#1\@nnil#1\@glsxtr@endescspch}%
4879     \fi
4880   \fi
4881   \@glsxtr@checkspch
4882 }
```

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

```
4883 \renewcommand*{\Glossentrydesc}[1]{%
4884   \glsdoifexistsorwarn{#1}%
4885   {%
4886     \glssetabbrvfmt{\glscategory{#1}}%
4887     \Glsaccessdesc{#1}%
4888   }%
4889 }
```

lossentrysymbol Redefine to set the abbreviation format and accessibility support.

```
4890 \renewcommand*{\lossentrysymbol}[1]{%
4891   \glsdoifexistsorwarn{#1}%
4892   {%
4893     \glssetabbrvfmt{\glscategory{#1}}%
4894     \glsaccesssymbol{#1}%
4895   }%
4896 }
```

lossentrysymbol Redefine to set the abbreviation format and accessibility support.

```
4897 \renewcommand*{\lossentrysymbol}[1]{%
4898   \glsdoifexistsorwarn{#1}%
4899   {%
```



```

4900 \glssetabbrvfmt{\glscategory{#1}}%
4901 \Glsaccesssymbol{#1}%
4902 }%
4903 }

```

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

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

```

4904 \newcommand*{\GlsXtrEnableInitialTagging}{%
4905 \ifstar\s@glxtr@enabletagging\@glxtr@enabletagging
4906 }
4907 \@onlypreamble\GlsXtrEnableInitialTagging

```

`\r@enabletagging` Starred version undefines command.

```

4908 \newcommand*{\s@glxtr@enabletagging}[2]{%
4909 \undef#2%
4910 \@glxtr@enabletagging{#1}{#2}%
4911 }

```

`\r@enabletagging` Internal command.

```

4912 \newcommand*{\@glxtr@enabletagging}[2]{%
    Set attributes for categories given in the first argument.
4913 \@for\@glxtr@cat:=#1\do
4914 {%
4915 \ifdefempty\@glxtr@cat
4916 {}%
4917 {\glssetcategoryattribute{\@glxtr@cat}{tagging}{true}}%
4918 }%
4919 \newrobustcmd*#2[1]{##1}%
4920 \def\@glxtr@taggingcs{#2}%
4921 \renewcommand*\@glxtr@activate@initialtagging{%
4922 \let#2\@glxtr@tag
4923 }%
4924 \ifundef\@gls@preglossaryhook
4925 {\GlossariesExtraWarning{Initial tagging requires at least
4926 glossaries.sty v4.19 to work correctly}}%
4927 {}%
4928 }

```

Are we using an old version of `mfirstuc` that has a bug in `\capitalisewords`? If so, patch it so we don't have a problem with a combination of tagging and title case.

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

```

4929 \ifundef\mfu@checkword@do
4930 {
4931 \newcommand*{\mfu@checkword@do}[1]{%

```

```

4932 \ifdefstring{\mfu@checkword@arg}{#1}%
4933 {%
4934     \let\@mfu@domakefirstuc\@firstofone
4935     \listbreak
4936 }%
4937 {}%
4938 }

```

`\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.

```

4939 \ifundef\mfu@checkword
4940 {
4941     \newcommand{\@glxtr@do@titlecaps@warn}{%
4942         \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
4943             support not available}%
4944         \let\@glxtr@do@titlecaps@warn\relax
4945     }
4946 }
4947 {
4948     \renewcommand*{\mfu@checkword}[1]{%
4949         \def\mfu@checkword@arg{#1}%
4950         \let\@mfu@domakefirstuc\makefirstuc
4951         \forlistloop\mfu@checkword@do\@mfu@nocaplist
4952     }
4953 }
4954 }
4955 {}% no patch required

```

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

```

4956 \newcommand*{\@glxtr@do@titlecaps@warn}{}

```

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

```

4957 \newcommand*{\@glxtr@activate@initialtagging}{}

```

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

```

4958 \newrobustcmd*{\@glxtr@tag}[1]{%
4959     \gl@ifattribute{\gl@currententrylabel}{tagging}{true}%
4960     {\glxtrtagfont{#1}}{#1}%
4961 }

```

`\glxtrtagfont` Used in the glossary.

```

4962 \newcommand*{\glxtrtagfont}[1]{\underline{#1}}

```

`preglossaryhook` This macro was introduced in `glossaries` version 4.19, so it may not be defined. If it hasn't been defined this feature is unavailable. A check is added for the entry's existence to prevent errors from occurring if the user removes an entry or changes the label, which can interrupt the build process.

```
4963 \ifdef\@gls@preglossaryhook
```

```
4964 {
```

```
4965   \renewcommand*{\@gls@preglossaryhook}{%
```

```
4966     \@glstr@activate@initialtagging
```

Since the glossaries are automatically scoped, \@glstr@org@postdescription shouldn't already be defined, but check anyway just as a precautionary measure.

```
4967   \ifundef\@glstr@org@postdescription
```

```
4968   {%
```

```
4969     \let\@glstr@org@postdescription\glspostdescription
```

```
4970     \renewcommand*{\glspostdescription}{%
```

```
4971       \ifglstryexists{\glscurrententrylabel}%
```

```
4972       {%
```

```
4973         \glstrpostdescription
```

```
4974         \@glstr@org@postdescription
```

```
4975       }%
```

```
4976     {}%
```

```
4977   }%
```

```
4978 }%
```

```
4979 {}%
```

Enable the options used by \@@glstrp:

```
4980   \glossxtrsetpopts
```

```
4981 }%
```

```
4982 }
```

```
4983 {}
```

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.

```
4984 \newcommand*{\glstrpostdescription}{%
```

```
4985   \csuse{glstrpostdesc\glscategory{\glscurrententrylabel}}%
```

```
4986 }
```

postdescgeneral

```
4987 \newcommand*{\glstrpostdescgeneral}{{}
```

xtrpostdescterm

```
4988 \newcommand*{\glstrpostdescterm}{{}
```

postdescacronym

```
4989 \newcommand*{\glstrpostdescacronym}{{}
```

escabbreviation

```
4990 \newcommand*{\glstrpostdescabbreviation}{{}
```

glspostlinkhook Redefine the post link hook used by commands like \gls to make it easier for categories or attributes to modify this action. Since this hook occurs outside the existence check of

commands like `\gls`, this needs to be checked again here. Do nothing if the entry hasn't been defined.

```
4991 \renewcommand*{\glspostlinkhook}{%
4992   \ifglentryexists{\glslabel}{\glstrpostlinkhook}{}%
4993 }
```

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

```
4994 \newcommand*{\glstrpostlinkhook}{%
4995   \glstrdiscardperiod{\glslabel}%
4996   {\glstrpostlinkendsentence}%
4997   {\glstrpostlink}%
4998 }
```

`\glstrpostlink`

```
4999 \newcommand*{\glstrpostlink}{%
5000   \csuse{glstrpostlink\glscategory{\glslabel}}%
5001 }
```

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

```
5002 \newcommand*{\glstrpostlinkendsentence}{%
5003   \ifcsdef{glstrpostlink\glscategory{\glslabel}}
5004   {%
5005     \csuse{glstrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
5006   .\spacefactor\sfcode'\. \relax
5007   }%
5008   {%
```

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

```
5009   \spacefactor\sfcode'\. \relax
5010   }%
5011 }
```

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

```
5012 \newcommand*{\glstrpostlinkAddDescOnFirstUse}{%
5013   \glstrifwasfirstuse{\space(\glssaccessdesc{\glslabel})}{}%
5014 }
```

`\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.

```
5015 \newcommand*{\glstrpostlinkAddSymbolOnFirstUse}{%
5016   \glstrifwasfirstuse
5017   {%
5018     \ifglshassymbol{\glslabel}{\space(\glssaccesssymbol{\glslabel})}{}%
5019   }%
5020   {}%
5021 }
```

`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).

```

5022 \newcommand*{\glxtrdiscardperiod}[3]{%
5023   \glxtrifwasfirstuse
5024   {%
5025     \glusifattribute{#1}{retainfirstuseperiod}{true}%
5026     {#3}%
5027     {%
5028       \glusifattribute{#1}{discardperiod}{true}%
5029       {%
5030         \glusifplural
5031         {%
5032           \glusifattribute{#1}{pluraldiscardperiod}{true}%
5033           {\glxtrifperiod{#2}{#3}}%
5034           {#3}%
5035         }%
5036         {%
5037           \glxtrifperiod{#2}{#3}%
5038         }%
5039       }%
5040     }%
5041   }%
5042 }%
5043 {%
5044   \glusifattribute{#1}{discardperiod}{true}%
5045   {%
5046     \glusifplural
5047     {%
5048       \glusifattribute{#1}{pluraldiscardperiod}{true}%
5049       {\glxtrifperiod{#2}{#3}}%
5050       {#3}%
5051     }%
5052     {%
5053       \glxtrifperiod{#2}{#3}%
5054     }%
5055   }%
5056   {#3}%
5057 }%
5058 }

```

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

```

5059 \newcommand*{\glxtrifperiod}[1]{\new@ifnextchar.\@firstoftwo{#1}}

```

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

`glxtr@punclist` List of characters identified as punctuation marks. (Be careful of babel shorthands!) This

doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```
5060 \newcommand*{\glxtr@punclist}{.,:;!}
```

punctuationmark Add character to punctuation list.

```
5061 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}
```

punctuationmarks Reset the punctuation list.

```
5062 \newcommand*{\glxtrsetpunctuationmarks}[1]{\def\glxtr@punclist{#1}}
```

```
\glxtrifpunc \glxtrifnextpunc{<true part>}{<false part>}
```

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

```
5063 \newcommand*{\glxtrifnextpunc}[2]{%
5064   \def\reserved@a{#1}%
5065   \def\reserved@b{#2}%
5066   \futurelet\@glspunc@token\glxtr@ifnextpunc
5067 }
```

glxtr@ifnextpunc

```
5068 \newcommand*{\glxtr@ifnextpunc}{%
5069   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
5070   \reserved@b
5071 }
```

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

```
5072 \newcommand*{\glxtr@ifpunctoken}[1]{%
5073   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
5074 }
```

glxtr@ifpunctoken

```
5075 \def\@glxtr@ifpunctoken#1#2{%
5076   \let\reserved@d=#2%
5077   \ifx\reserved@d\@nnil
5078     \let\glxtr@next\@glxtr@notfoundinlist
5079   \else
5080     \ifx#1\reserved@d
5081       \let\glxtr@next\@glxtr@foundinlist
5082     \else
5083       \let\glxtr@next\@glxtr@ifpunctoken
5084     \fi
5085   \fi
5086   \glxtr@next#1%
5087 }
```

glxtr@foundinlist

```
5088 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
5089 \def\@glxstr@notfoundinlist#1{\@secondoftwo}
```

glxstrdopostpunc

```
\glxstrdopostpunc{<code>}
```

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

```
5090 \newcommand{\glxstrdopostpunc}[1]{%
5091   \glxstrifnextpunc{\@glxstr@swaptwo{#1}}{#1}%
5092 }
```

@glxstr@swaptwo

```
5093 \newcommand{\@glxstr@swaptwo}[2]{#2#1}
```

1.6 Abbreviations

The “acronym” code from glossaries is misnamed as it’s more often used for other forms of abbreviations. This code corrects this inconsistency, but rather than just having synonyms, provide commands for abbreviations that have a similar, but not identical, underlying mechanism to acronyms.

If there’s a style for the given category, apply it.

```
5094 \define@key{glxstrabbrv}{category}{%
5095   \edef\glscategorylabel{#1}%
5096   \ifcsdef{@glxabbrv@current@#1}%
5097   {%
```

Warning should already have been issued.

```
5098   \let\@glxstr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
5099   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
5100   \glxstr@applyabbrvstyle{\csname @glxabbrv@current@#1\endcsname}%
5101   \let\GlsXtrWarnDeprecatedAbbrStyle\@glxstr@orgwarndep
5102 }%
5103 {}%
5104 }
```

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

```
5105 \define@key{glxstrabbrv}{shortplural}{%
5106   \def\@glx@shortpl{#1}%
5107 }
```

Similarly for the long plural form.

```
5108 \define@key{glxstrabbrv}{longplural}{%
5109   \def\@glx@longpl{#1}%
5110 }
```

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

`\glsshortpltok`

```
5111 \newtoks\glsshortpltok
```

`\glslongpltok`

```
5112 \newtoks\glslongpltok
```

`\glsxtr@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.

```
5113 \newcommand*{\@glsxtr@insertdots}[2]{%
5114   \def#1{}%
5115   \@glsxtr@insert@dots#1#2\@nnil
5116 }
```

`\glsxtr@insert@dots`

```
5117 \newcommand*{\@glsxtr@insert@dots}[2]{%
5118   \ifx\@nnil#2\relax
5119   \let\@glsxtr@insert@dots@next\@gobble
5120   \else
5121   \ifx\relax#2\relax
5122   \else
5123     \appto#1{#2.}%
5124   \fi
5125   \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
5126   \fi
5127   \@glsxtr@insert@dots@next#1%
5128 }
```

Similarly provide a way of replacing spaces with `\glsxtrwordsep`, which first needs to be defined:

`\glsxtrwordsep`

```
5129 \newcommand*{\glsxtrwordsep}{\space}
```

Each word is marked with

`\glsxtrword`

```
5130 \newcommand*{\glsxtrword}[1]{#1}
```

`\glsxtr@markwordseps`

```
5131 \newcommand*{\@glsxtr@markwordseps}[2]{%
5132   \def#1{}%
5133   \@glsxtr@mark@wordseps#1#2 \@nnil
5134 }
```


r@mark@wordseps

```
5135 \def\@glxstr@mark@wordseps#1#2 #3{%
5136   \ifdefempty{#1}%
5137   {\def#1{\protect\glxstrword{#2}}}%
5138   {\appto#1{\protect\glxstrwordsep\protect\glxstrword{#2}}}%
5139   \ifx\@nnil#3\relax
5140   \let\@glxstr@mark@wordseps@next\relax
5141   \else
5142   \def\@glxstr@mark@wordseps@next{%
5143     \@glxstr@mark@wordseps#1#3}%
5144   \fi
5145   \@glxstr@mark@wordseps@next
5146 }
```

newabbreviation Define a new generic abbreviation.

```
5147 \newcommand*{\newabbreviation}[4] [] {%
5148   \glxstr@newabbreviation{#1}{#2}{#3}{#4}%
5149 }
```

newabbreviation Internal macro. (bib2gls has an option that needs to temporarily redefine \newabbreviation. This is just makes it easier to save and restore the original definition.)

```
5150 \newcommand*{\glxstr@newabbreviation}[4] {%
5151   \glskeylisttok{#1}%
5152   \glslabeltok{#2}%
5153   \glsshorttok{#3}%
5154   \glslongtok{#4}%

```

Save the original short and long values (before attribute settings modify them).

```
5155   \def\glxstrorgshort{#3}%
5156   \def\glxstrorglong{#4}%

```

Get the category.

```
5157   \def\glscategorylabel{abbreviation}%
5158   \glxstr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%

```

Ignore the shortplural and longplural keys.

```
5159   \setkeys*{\glxstrabbrv}[shortplural,longplural]{#1}%

```

Set the default long plural

```
5160   \def\@gls@longpl{#4\glspluralsuffix}%
5161   \let\@gls@default@longpl\@gls@longpl

```

Has the markwords attribute been set?

```
5162   \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
5163   {%
5164     \@glxstr@markwordseps\@gls@long{#4}%
5165     \expandafter\def\expandafter\@gls@longpl\expandafter
5166       {\@gls@long\glspluralsuffix}%
5167     \let\@gls@default@longpl\@gls@longpl

```

Update \glslongtok.

```

5168   \expandafter\glslongtok\expandafter{\@gls@long}%
5169   }%
5170   {}%

Has the markshortwords attribute been set? (Not compatible with insertdots.)
5171   \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%
5172   {%
5173     \@glsxtr@markwordseps\@gls@short{#3}%
5174   }%
5175   {%

Has the insertdots attribute been set?
5176   \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
5177   {%
5178     \@glsxtr@insertdots\@gls@short{#3}%
5179     \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
5180   }%
5181   {\def\@gls@short{#3}}%
5182   }%

Has the aposplural attribute been set? (Not compatible with noshortplural.)
5183   \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
5184   {%
5185     \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
5186       '\abbrvpluralsuffix}%
5187   }%
5188   {%

Has the noshortplural attribute been set?
5189   \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
5190   {%
5191     \let\@gls@shortpl\@gls@short
5192   }%
5193   {%
5194     \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
5195       \abbrvpluralsuffix}%
5196   }%
5197   }%

Update \glsshorttok:
5198   \expandafter\glsshorttok\expandafter{\@gls@short}%

Hook for further customisation if required:
5199   \glsxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%

Get the short and long plurals provided by user in optional argument to override defaults, if
necessary. Ignore the category key (already obtained).
5200   \setkeys*{glsxtrabbrv}[category]{#1}%

Has the plural been explicitly set?
5201   \ifx\@gls@default@longpl\@gls@longpl
5202   \else

```

Has the markwords attribute been set?

```

5203 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
5204 {%
5205 \expandafter\@glstr@markwordseps\expandafter\@gls@longpl\expandafter
5206 {\@gls@longpl}%
5207 }%
5208 {}%
5209 \fi

```

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

```

5210 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
5211 \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

5212 \newabbreviationhook

```

Define this entry:

```

5213 \protected@edef\@do@newglossaryentry{%
5214 \noexpand\newglossaryentry{\the\glslabeltok}%
5215 {%
5216 type=\glstrabbrvtype,%
5217 category=abbreviation,%
5218 short={\the\glsshorttok},%
5219 shortplural={\the\glsshortpltok},%
5220 long={\the\glslongtok},%
5221 longplural={\the\glslongpltok},%
5222 name={\the\glsshorttok},%
5223 \CustomAbbreviationFields,%
5224 \the\glskeylisttok
5225 }%
5226 }%
5227 \@do@newglossaryentry
5228 \GlsXtrPostNewAbbreviation
5229 }

```

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

```

5230 \newcommand*{\glstrnewabbrevpresetkeyhook}[3]{}

```

`\NewAbbreviation` Hook used by abbreviation styles.

```

5231 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

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

```

5232 \newcommand*{\newabbreviationhook}{}

```

`\reviationFields`

```

5233 \newcommand*{\CustomAbbreviationFields}{}

```

`\glstrparen` For the parenthetical styles.

```

5234 \newcommand*{\glstrparen}[1]{(#1)}

```

`\lsxtrfullformat` Full format without case change.

```

5235 \newcommand*{\lsxtrfullformat}[2]{%
5236   \glsfirstlongfont{\glsaccesslong{#1}}#2\lsxtrfullsep{#1}%
5237   \lsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
5238 }
```

`\lsxtrfullformat` Full format with case change.

```

5239 \newcommand*{\Glsxtrfullformat}[2]{%
5240   \glsfirstlongfont{\Glsaccesslong{#1}}#2\lsxtrfullsep{#1}%
5241   \lsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
5242 }
```

`\xtrfullplformat` Plural full format without case change.

```

5243 \newcommand*{\xtrfullplformat}[2]{%
5244   \glsfirstlongfont{\glsaccesslongpl{#1}}#2\lsxtrfullsep{#1}%
5245   \lsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}}}%
5246 }
```

`\xtrfullplformat` Plural full format with case change.

```

5247 \newcommand*{\Glxtrfullplformat}[2]{%
5248   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\lsxtrfullsep{#1}%
5249   \lsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}}}%
5250 }
```

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

```

5251 \newcommand*{\glsxtrfullsep}[1]{\space}
```

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

`\inlinefullformat` Full format without case change.

```

5252 \newcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}
```

`\inlinefullformat` Full format with case change.

```

5253 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}
```

`\xtrfullplformat` Plural full format without case change.

```

5254 \newcommand*{\xtrinlinefullplformat}{\glsxtrfullplformat}
```

`\inefullplformat` Plural full format with case change.

```

5255 \newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}
```

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`

```

5256 \renewcommand*{\glsentryfull}[1]{\glsxtrinlinefullformat{#1}{}}
```

`\Glsentryfull`
5257 `\renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}`

`\glsentryfullpl`
5258 `\renewcommand*{\glsentryfullpl}[1]{\glsxtrinlinefullplformat{#1}{}}`

`\Glsentryfullpl`
5259 `\renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}`

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.
5260 `\newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.
5261 `\newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}`

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.
5262 `\newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont`
5263 `\newcommand*{\glsabbrvdefaultfont}[1]{#1}`

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.
5264 `\newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}`

`\glslongdefaultfont` Default font changing command used for the long form in commands like `\glsxtrlong`.
5265 `\newcommand*{\glslongdefaultfont}[1]{#1}`

`\glsfirstlongfont` Font changing command used for the long form on first use or in the full format.
5266 `\newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}`

`\glsfirstlongdefaultfont`
5267 `\newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}`

`\glsbrvpluralsuffix` Default plural suffix. Allow an alternative default suffix for abbreviations.
5268 `\newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}`

`\glsbrvpluralsuffix` Default plural suffix.
5269 `\newcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}`

`\glsxtrfull` Full form (no case-change).
5270 `\newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\ns@glsxtrfull}`
5271 `\newcommand*{\ns@glsxtrfull}[2][{}]{%`
5272 `\new@ifnextchar[{\@glsxtr@full{#1}{#2}}{%`
5273 `{\@glsxtr@full{#1}{#2}[]}%`
5274 `}`

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

```
5275 \def\@glxtr@full#1#2[#3]{%
5276   \glsdoifexists{#2}%
5277   {%
5278     \glsetabbrvfmt{\glscategory{#2}}%
5279     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5280     \let\glsifplural\@secondoftwo
5281     \let\glscapscase\@firstofthree
5282     \let\glsinsert\@empty
5283     \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.

```
5284   \glxtrsetupfulldefs
5285   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5286   }%
5287   \glspostlinkhook
5288 }
```

`trsetupfulldefs`

```
5289 \newcommand*{\glxtrsetupfulldefs}{%
5290   \let\glxtrifwasfirstuse\@firstoftwo
5291 }
```

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

```
5292 \newrobustcmd*{\Glsxtrfull}{\@gl@hyp@opt\@ns@Glsxtrfull}
5293 \newcommand*\ns@Glsxtrfull[2][ ]{%
5294   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}%
5295   {\@Glsxtr@full{#1}{#2} [ ]}%
5296 }
```

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

```
5297 \def\@Glsxtr@full#1#2[#3]{%
5298   \glsdoifexists{#2}%
5299   {%
5300     \glsetabbrvfmt{\glscategory{#2}}%
5301     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5302     \let\glsifplural\@secondoftwo
5303     \let\glscapscase\@secondofthree
5304     \let\glsinsert\@empty
5305     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
5306     \glxtrsetupfulldefs
5307     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5308     }%
5309     \glspostlinkhook
5310 }
```

`\GLSxtrfull` Full form (all uppercase).

```
5311 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\ns@GLSxtrfull}
5312 \newcommand*\ns@GLSxtrfull[2] [] {%
5313   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}}%
5314   {\@GLSxtr@full{#1}{#2} []}%
5315 }
```

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

```
5316 \def\@GLSxtr@full#1#2[#3] {%
5317   \glsdoifexists{#2}%
5318   {%
5319     \glssetabbrvfmt{\glscategory{#2}}%
5320     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5321     \let\glsifplural\@secondoftwo
5322     \let\glsapscase\@thirdofthree
5323     \let\glsinsert\@empty
5324     \def\glscustomtext{\mfirstucMakeUppercase{\glsxtrinlinefullformat{#2}{#3}}}%
5325     \glsxtrsetupfulldefs
5326     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5327   }%
5328   \glspostlinkhook
5329 }
```

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

```
5330 \newrobustcmd*{\glsxtrfullpl}{\@gls@hyp@opt\ns@glsxtrfullpl}
5331 \newcommand*\ns@glsxtrfullpl[2] [] {%
5332   \new@ifnextchar[{\@glsxtr@fullpl{#1}{#2}}}%
5333   {\@glsxtr@fullpl{#1}{#2} []}%
5334 }
```

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

```
5335 \def\@glsxtr@fullpl#1#2[#3] {%
5336   \glsdoifexists{#2}%
5337   {%
5338     \glssetabbrvfmt{\glscategory{#2}}%
5339     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5340     \let\glsifplural\@firstoftwo
5341     \let\glsapscase\@firstofthree
5342     \let\glsinsert\@empty
5343     \def\glscustomtext{\glsxtrinlinefullplformat{#2}{#3}}%
5344     \glsxtrsetupfulldefs
5345     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5346   }%
5347   \glspostlinkhook
5348 }
```

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

```
5349 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\ns@Glsxtrfullpl}
5350 \newcommand*\ns@Glsxtrfullpl[2] [] {%
```

```

5351 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}}%
5352           {\@Glsxtr@fullpl{#1}{#2}[]}%
5353 }

```

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

```

5354 \def\@Glsxtr@fullpl#1#2[#3]{%
5355   \glsdoifexists{#2}%
5356   {%
5357     \glssetabbrvfmt{\glscategory{#2}}%
5358     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5359     \let\glsifplural\@firstoftwo
5360     \let\glscapscase\@secondofthree
5361     \let\glsinsert\@empty
5362     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
5363     \glsxtrsetupfulldefs
5364     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5365   }%
5366   \glspostlinkhook
5367 }

```

`\Glsxtrfullpl` Plural full form (all upper case).

```

5368 \newrobustcmd*{\Glsxtrfullpl}{\@gl@hyp@opt\@ns@Glsxtrfullpl}
5369 \newcommand*\ns@Glsxtrfullpl[2][]{%
5370   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}}%
5371   {\@Glsxtr@fullpl{#1}{#2}[]}%
5372 }

```

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

```

5373 \def\@Glsxtr@fullpl#1#2[#3]{%
5374   \glsdoifexists{#2}%
5375   {%
5376     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5377     \let\glsifplural\@firstoftwo
5378     \let\glscapscase\@thirdofthree
5379     \let\glsinsert\@empty
5380     \def\glscustomtext{%
5381       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
5382     \glsxtrsetupfulldefs
5383     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5384   }%
5385   \glspostlinkhook
5386 }

```

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

`\glsxtrshort`

```

5387 \newrobustcmd*{\glsxtrshort}{\@gl@hyp@opt\@ns@glsxtrshort}

```

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


```

5388 \newcommand*{\ns@glstrshort}[2] [] {%
5389   \new@ifnextchar[{\@glstrshort{#1}{#2}}{\@glstrshort{#1}{#2} []}%
5390 }

```

Read in the final optional argument:

```

5391 \def\@glstrshort#1#2[#3] {%
5392   \glstoifexists{#2}%
5393   {%

```

Need to make sure \glabbrvfont is set correctly.

```

5394     \glsetabbrvfmt{\glscategory{#2}}%
5395     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5396     \let\glstrifwasfirstuse\@secondoftwo
5397     \let\gl@ifplural\@secondoftwo
5398     \let\glscapscase\@firstofthree
5399     \let\glinsert\@empty
5400     \def\glscustomtext{%
5401       \glabbrvfont{\glaccessshort{#2}\ifglstrinsertinside#3\fi}%
5402       \ifglstrinsertinside\else#3\fi
5403     }%
5404     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5405   }%
5406   \glspostlinkhook
5407 }

```

\Glsxtrshort

```

5408 \newrobustcmd*{\Glsxtrshort}{\@gl@hyp@opt\ns@Glsxtrshort}

```

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

```

5409 \newcommand*{\ns@Glsxtrshort}[2] [] {%
5410   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} []}%
5411 }

```

Read in the final optional argument:

```

5412 \def\@Glsxtrshort#1#2[#3] {%
5413   \glstoifexists{#2}%
5414   {%
5415     \glsetabbrvfmt{\glscategory{#2}}%
5416     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5417     \let\glstrifwasfirstuse\@secondoftwo
5418     \let\gl@ifplural\@secondoftwo
5419     \let\glscapscase\@secondofthree
5420     \let\glinsert\@empty
5421     \def\glscustomtext{%
5422       \glabbrvfont{\Glsaccessshort{#2}\ifglstrinsertinside#3\fi}%
5423       \ifglstrinsertinside\else#3\fi
5424     }%
5425     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5426   }%
5427   \glspostlinkhook
5428 }

```

\GLSxtrshort

```
5429 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\ns@GLSxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
5430 \newcommand*{\ns@GLSxtrshort}[2] [] {%
5431   \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} []}]%
5432 }

    Read in the final optional argument:
5433 \def\@GLSxtrshort#1#2[#3] {%
5434   \glsdoifexists{#2}%
5435   {%
5436     \glssetabbrvfmt{\glscategory{#2}}%
5437     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5438     \let\glsxtrifwasfirstuse\@secondoftwo
5439     \let\glsifplural\@secondoftwo
5440     \let\glsapscase\@thirdofthree
5441     \let\glsinsert\@empty
5442     \def\glscustomtext{%
5443       \mfirstucMakeUppercase
5444       {\glsabbrvfont{\glsaccessshort{#2}}\ifglsxtrininsertinside#3\fi}%
5445       \ifglsxtrininsertinside\else#3\fi
5446     }%
5447   }%
5448   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5449   }%
5450   \glspostlinkhook
5451 }
```

\glsxtrlong

```
5452 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
5453 \newcommand*{\ns@glsxtrlong}[2] [] {%
5454   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2} []}]%
5455 }

    Read in the final optional argument:
5456 \def\@glsxtrlong#1#2[#3] {%
5457   \glsdoifexists{#2}%
5458   {%
5459     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5460     \let\glsxtrifwasfirstuse\@secondoftwo
5461     \let\glsifplural\@secondoftwo
5462     \let\glsapscase\@firstofthree
5463     \let\glsinsert\@empty
5464     \def\glscustomtext{%
5465       \glsfont{\glsaccesslong{#2}}\ifglsxtrininsertinside#3\fi}%
5466       \ifglsxtrininsertinside\else#3\fi
5467     }%
5468     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5469   }
```

```

5469 }%
5470 \glspostlinkhook
5471 }

```

\Glsxtrlong

```

5472 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
5473 \newcommand*{\ns@Glsxtrlong}[2] [] {%
5474   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
5475 }

    Read in the final optional argument:
5476 \def\@Glsxtrlong#1#2[#3]{%
5477   \glsdoifexists{#2}%
5478   {%
5479     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5480     \let\glstrifwasfirstuse\@secondoftwo
5481     \let\glsifplural\@secondoftwo
5482     \let\glscapscase\@thirdofthree
5483     \let\glsinsert\@empty
5484     \def\glscustomtext{%
5485       \glslongfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
5486       \ifglstrinsertinside\else#3\fi
5487     }%
5488     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5489   }%
5490   \glspostlinkhook
5491 }

```

\GLSxtrlong

```

5492 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\@ns@GLSxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
5493 \newcommand*{\ns@GLSxtrlong}[2] [] {%
5494   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
5495 }

    Read in the final optional argument:
5496 \def\@GLSxtrlong#1#2[#3]{%
5497   \glsdoifexists{#2}%
5498   {%
5499     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5500     \let\glstrifwasfirstuse\@secondoftwo
5501     \let\glsifplural\@secondoftwo
5502     \let\glscapscase\@thirdofthree
5503     \let\glsinsert\@empty
5504     \def\glscustomtext{%
5505       \mfirstucMakeUppercase
5506       {\glslongfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
5507       \ifglstrinsertinside\else#3\fi

```

```

5508     }%
5509     }%
5510     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5511     }%
5512     \glspostlinkhook
5513 }

```

Plural short forms:

\glsxtrshortpl

```

5514 \newrobustcmd*{\glsxtrshortpl}{\@gls@hyp@opt\@ns@glsxtrshortpl}

```

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

```

5515 \newcommand*{\ns@glsxtrshortpl}[2] [] {%
5516   \new@ifnextchar[{\@glsxtrshortpl{#1}{#2}}{\@glsxtrshortpl{#1}{#2} []}%
5517 }

```

Read in the final optional argument:

```

5518 \def\@glsxtrshortpl#1#2[#3] {%
5519   \glsdoifexists{#2}%
5520   {%
5521     \glssetabbrvfmt{\glscategory{#2}}%
5522     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5523     \let\glsxtrifwasfirstuse\@secondoftwo
5524     \let\glsifplural\@firstoftwo
5525     \let\glscapscase\@firstofthree
5526     \let\glsinsert\@empty
5527     \def\glscustomtext{%
5528       \glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
5529       \ifglsxtrininsertinside\else#3\fi
5530     }%
5531     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5532     }%
5533     \glspostlinkhook
5534 }

```

\Glsxtrshortpl

```

5535 \newrobustcmd*{\Glsxtrshortpl}{\@gls@hyp@opt\@ns@Glsxtrshortpl}

```

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

```

5536 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
5537   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
5538 }

```

Read in the final optional argument:

```

5539 \def\@Glsxtrshortpl#1#2[#3] {%
5540   \glsdoifexists{#2}%
5541   {%
5542     \glssetabbrvfmt{\glscategory{#2}}%
5543     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5544     \let\glsxtrifwasfirstuse\@secondoftwo

```

```

5545 \let\glsifplural\@firstoftwo
5546 \let\glscapscase\@secondofthree
5547 \let\glsinsert\@empty
5548 \def\glscustomtext{%
5549 \glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrinsertinside#3\fi}%
5550 \ifglsxtrinsertinside\else#3\fi
5551 }%
5552 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5553 }%
5554 \glspostlinkhook
5555 }

```

`\GLSxtrshortpl`

```

5556 \newrobustcmd*{\GLSxtrshortpl}{\@gls@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
5557 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
5558 \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
5559 }

```

Read in the final optional argument:

```

5560 \def\@GLSxtrshortpl#1#2[#3] {%
5561 \glsdoifexists{#2}%
5562 {%
5563 \glssetabbrvfmt{\glscategory{#2}}%
5564 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5565 \let\glsxtrifwasfirstuse\@secondoftwo
5566 \let\glsifplural\@firstoftwo
5567 \let\glsapspace\@thirdofthree
5568 \let\glsinsert\@empty
5569 \def\glscustomtext{%
5570 \mfirstucMakeUppercase
5571 {\glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrinsertinside#3\fi}%
5572 \ifglsxtrinsertinside\else#3\fi
5573 }%
5574 }%
5575 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5576 }%
5577 \glspostlinkhook
5578 }

```

Plural long forms:

`\glsxtrlongpl`

```

5579 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\ns@glsxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
5580 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
5581 \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
5582 }

```

Read in the final optional argument:

```
5583 \def\@glxstrlongpl#1#2[#3]{%
5584   \glstoifexists{#2}%
5585   {%
5586     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5587     \let\glxstrifwasfirstuse\@secondoftwo
5588     \let\gl@ifplural\@firstoftwo
5589     \let\glscapscase\@firstofthree
5590     \let\glinsert\@empty
5591     \def\glscustomtext{%
5592       \glslongfont{\glaccesslongpl{#2}\ifglxstrinsertinside#3\fi}%
5593       \ifglxstrinsertinside\else#3\fi
5594     }%
5595     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5596   }%
5597   \glspostlinkhook
5598 }
```

\Glsxtrlongpl

```
5599 \newrobustcmd*{\Glsxtrlongpl}{\@gl@hyp@opt\ns@Glsxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
5600 \newcommand*{\ns@Glsxtrlongpl}[2][]{%
5601   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2}[]}%
5602 }
```

Read in the final optional argument:

```
5603 \def\@Glsxtrlongpl#1#2[#3]{%
5604   \glstoifexists{#2}%
5605   {%
5606     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5607     \let\glxstrifwasfirstuse\@secondoftwo
5608     \let\gl@ifplural\@firstoftwo
5609     \let\glscapscase\@secondofthree
5610     \let\glinsert\@empty
5611     \def\glscustomtext{%
5612       \glslongfont{\Glsaccesslongpl{#2}\ifglxstrinsertinside#3\fi}%
5613       \ifglxstrinsertinside\else#3\fi
5614     }%
5615     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5616   }%
5617   \glspostlinkhook
5618 }
```

\GLSxtrlongpl

```
5619 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\ns@GLSxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
5620 \newcommand*{\ns@GLSxtrlongpl}[2][]{%
5621   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2}[]}%
5622 }
```

Read in the final optional argument:

```

5623 \def\@GLSxtrlongpl#1#2[#3]{%
5624   \glsdoifexists{#2}%
5625   {%
5626     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
5627     \let\glxtrifwasfirstuse\@secondoftwo
5628     \let\glsifplural\@firstoftwo
5629     \let\glscapscase\@thirdofthree
5630     \let\glsinsert\@empty
5631     \def\glscustomtext{%
5632       \mfirstucMakeUppercase
5633       {\glslongfont{\glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
5634       \ifglxtrininsertinside\else#3\fi
5635     }%
5636   }%
5637   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5638 }%
5639 \glspostlinkhook
5640 }

```

`\glssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

5641 \newcommand*{\glssetabbrvfmt}[1]{%
5642   \ifcsdef{@glsabbrv@current@#1}%
5643   {\glxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
5644   {\glxtr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
5645 }

```

`sxtrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

5646 \newcommand*{\glxtrgenabbrvfmt}{%
5647   \ifdefempty\glscustomtext
5648   {%
5649     \ifgl@sused\glslabel
5650     {%

```

Subsequent use:

```

5651     \glsifplural
5652     {%

```

Subsequent plural form:

```

5653     \glscapscase
5654     {%

```

Subsequent plural form, don't adjust case:

```

5655     \glxtrsubsequentplfmt{\glslabel}{\glsinsert}%
5656     }%
5657     {%

```

Subsequent plural form, make first letter upper case:

```

5658     \Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%
5659     }%
5660     {%

```

Subsequent plural form, all caps:

```
5661      \mfirstucMakeUppercase
5662      {\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%
5663      }%
5664      }%
5665      {%
```

Subsequent singular form

```
5666      \glscapscase
5667      {%
```

Subsequent singular form, don't adjust case:

```
5668      \glxtrsubsequentfmt{\glslabel}{\glsinsert}%
5669      }%
5670      {%
```

Subsequent singular form, make first letter upper case:

```
5671      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%
5672      }%
5673      {%
```

Subsequent singular form, all caps:

```
5674      \mfirstucMakeUppercase
5675      {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%
5676      }%
5677      }%
5678      }%
5679      {%
```

First use:

```
5680      \glsifplural
5681      {%
```

First use plural form:

```
5682      \glscapscase
5683      {%
```

First use plural form, don't adjust case:

```
5684      \glxtrfullplformat{\glslabel}{\glsinsert}%
5685      }%
5686      {%
```

First use plural form, make first letter upper case:

```
5687      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
5688      }%
5689      {%
```

First use plural form, all caps:

```
5690      \mfirstucMakeUppercase
5691      {\glxtrfullplformat{\glslabel}{\glsinsert}}%
5692      }%
5693      }%
5694      {%
```


First use singular form

```
5695      \glscapscase
5696      {%
```

First use singular form, don't adjust case:

```
5697      \glxtrfullformat{\glslabel}{\glsinsert}%
5698      }%
5699      {%
```

First use singular form, make first letter upper case:

```
5700      \Glsxtrfullformat{\glslabel}{\glsinsert}%
5701      }%
5702      {%
```

First use singular form, all caps:

```
5703      \mfirstucMakeUppercase
5704      {\glxtrfullformat{\glslabel}{\glsinsert}}%
5705      }%
5706      }%
5707      }%
5708      }%
5709      {%
```

User supplied text.

```
5710      \glscustomtext
5711      }%
5712 }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
5713 \newcommand*{\glxtrsubsequentfmt}[2]{%
5714   \glsabbrvfont{\glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
5715   \ifglxtrininsertinside \else#2\fi
5716 }
5717 \let\glxtrdefaultsubsequentfmt\glxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural no case change).

```
5718 \newcommand*{\glxtrsubsequentplfmt}[2]{%
5719   \glsabbrvfont{\glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
5720   \ifglxtrininsertinside \else#2\fi
5721 }
5722 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt
```

trsubsequentfmt Subsequent use format (singular, first letter uppercase).

```
5723 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
5724   \glsabbrvfont{\Glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
5725   \ifglxtrininsertinside \else#2\fi
5726 }
5727 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural, first letter uppercase).

```
5728 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
5729   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglxtrinsertinside #2\fi}%
5730   \ifglxtrinsertinside \else#2\fi
5731 }
5732 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt
```

1.6.1 Abbreviation Styles Setup

breviationstyle

```
5733 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
5734   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
5735   {%
5736     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
5737   }%
5738   {%
```

Have abbreviations already been defined for this category?

```
5739   \ifcsstring{@glsabbrv@current@#1}{#2}%
5740   {%
```

Style already set.

```
5741   }%
5742   {%
5743     \def\@glxtr@dostylewarn{%
5744       \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
5745       {%
5746         \def\@glxtr@dostylewarn{\GlossariesWarning{Abbreviation
5747           style has been switched \MessageBreak
5748           for category ‘#1’, \MessageBreak
5749           but there have already been entries \MessageBreak
5750           defined for this category. Unwanted \MessageBreak
5751           side-effects may result}}%
5752       \@endfortrue
5753     }%
5754     \@glxtr@dostylewarn
```

Set up the style for the given category.

```
5755     \csdef{@glsabbrv@current@#1}{#2}%
5756     \glsxtr@applyabbrvstyle{#2}%
5757   }%
5758 }%
5759 }
```

applyabbrvstyle Apply the abbreviation style without existence check.

```
5760 \newcommand*{\glxtr@applyabbrvstyle}[1]{%
5761   \csuse{@glsabbrv@dispstyle@setup@#1}%
5762   \csuse{@glsabbrv@dispstyle@fmts@#1}%
5763 }
```

r@applyabbrvfmt Only apply the style formats.

```
5764 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
5765   \csuse{@glsabbrv@dispstyle@fmts@#1}%
5766 }
```

breivationstyle This is different from \newacronymstyle. The first argument is the label, the second argument sets the information required when defining the new abbreviation and the third argument sets the commands used to display the full format.

```
5767 \newcommand*{\newabbreviationstyle}[3]{%
5768   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
5769   {%
5770     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
5771       defined}{}%
5772   }%
5773   {%
5774     \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
5775     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
5776     #2}%
5777     \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
5778     \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
5779     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
5780     \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
5781     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

Reset \glxtrsubsequentfmt etc in case a style changes this.

```
5782     \let\glxtrsubsequentfmt\glxtrdefaultsubsequentfmt
5783     \let\glxtrsubsequentplfmt\glxtrdefaultsubsequentplfmt
5784     \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
5785     \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
5786     #3}%
5787   }%
5788 }
```

breivationstyle

```
5789 \newcommand*{\renewabbreviationstyle}[3]{%
5790   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
5791   {%
5792     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
5793   }%
5794   {%
5795     \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
5796     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
5797     #2}%
5798     \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```

5799 \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
5800 \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
5801 \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
5802 \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
5803   #3}%
5804 }%
5805 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

5806 \newcommand*{\letabbreviationstyle}[2]{%
5807   \csletcs{@glssabbrv@dispstyle@setup@#1}{@glssabbrv@dispstyle@setup@#2}%
5808   \csletcs{@glssabbrv@dispstyle@fmts@#1}{@glssabbrv@dispstyle@fmts@#2}%
5809 }

```

deprecated@abbrstyle `\@glxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

5810 \newcommand*{\@glxtr@deprecated@abbrstyle}[2]{%
5811   \csdef{@glssabbrv@dispstyle@setup@#1}{%
5812     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
5813     \csuse{@glssabbrv@dispstyle@setup@#2}%
5814   }%
5815   \csletcs{@glssabbrv@dispstyle@fmts@#1}{@glssabbrv@dispstyle@fmts@#2}%
5816 }

```

deprecatedAbbrStyle Generate warning for deprecated style use.

```

5817 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
5818   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
5819     use ‘#2’ instead}%
5820 }

```

UseAbbrStyleSetup

```

5821 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
5822   \ifcsundef{@glssabbrv@dispstyle@setup@#1}%
5823   {%
5824     \PackageError{glossaries-extra}%
5825       {Unknown abbreviation style definitions ‘#1’}{}%
5826   }%
5827   {%
5828     \csname @glssabbrv@dispstyle@setup@#1\endcsname
5829   }%
5830 }

```

seAbbrStyleFmts

```
5831 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
5832   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
5833   {%
5834     \PackageError{glossaries-extra}%
5835     {Unknown abbreviation style formats ‘#1’}{}%
5836   }%
5837   {%
5838     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
5839   }%
5840 }
```

1.6.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn’t set to “true”. If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glsxtrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

xtrinsertinside Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```
5841 \newif\ifglsxtrinsertinside
5842 \glsxtrinsertinsidefalse
```

long-short

```
5843 \newabbreviationstyle{long-short}%
5844 {%
5845   \renewcommand*{\CustomAbbreviationFields}{%
5846     name={\protect\glsabbrvfont{\the\glsshorttok}},
5847     sort={\the\glsshorttok},
5848     first={\protect\glsfirstlongfont{\the\glslongtok}}%
5849     \protect\glsxtrfullsep{\the\glslabeltok}%
5850     \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
5851     firstplural={\protect\glsfirstlongfont{\the\glslongtok}}%
5852     \protect\glsxtrfullsep{\the\glslabeltok}%
5853     \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
5854     plural={\protect\glsabbrvfont{\the\glsshortpltok}}},%
5855     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
5856 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5857   \glsattribute{\the\glslabeltok}{regular}%
5858   {%
```

```

5859 \glsetattribute{\the\glslabelltok}{regular}{false}%
5860 }%
5861 {}%
5862 }%
5863 }%
5864 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5865 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
5866 \renewcommand*\glabbrvfont[1]{\glabbrvdefaultfont{##1}}%
5867 \renewcommand*\glfirstabbrvfont[1]{\glfirstabbrvdefaultfont{##1}}%
5868 \renewcommand*\glfirstlongfont[1]{\glfirstlongdefaultfont{##1}}%
5869 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5870 \renewcommand*\glxtrfullformat[2]{%
5871 \glfirstlongfont{\glaccesslong{##1}\ifglxtrinsertinside##2\fi}%
5872 \ifglxtrinsertinside\else##2\fi
5873 \glxtrfullsep{##1}%
5874 \glxtrparen{\glfirstabbrvfont{\glaccessshort{##1}}}%
5875 }%
5876 \renewcommand*\glxtrfullplformat[2]{%
5877 \glfirstlongfont{\glaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5878 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5879 \glxtrparen{\glfirstabbrvfont{\glaccessshortpl{##1}}}%
5880 }%
5881 \renewcommand*\Glsxtrfullformat[2]{%
5882 \glfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
5883 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5884 \glxtrparen{\glfirstabbrvfont{\glaccessshort{##1}}}%
5885 }%
5886 \renewcommand*\Glsxtrfullplformat[2]{%
5887 \glfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5888 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
5889 \glxtrparen{\glfirstabbrvfont{\glaccessshortpl{##1}}}%
5890 }%
5891 }

```

Set this as the default style for general abbreviations:

```

5892 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

5893 \newcommand*\glxtrlongshortdescsort{%
5894 \expandonce\glxtrorglong\space (\expandonce\glxtrorgshort)%
5895 }

```

ngshortdescname

```

5896 \newcommand*\glxtrlongshortdescname{%
5897 \protect\glslongfont{\the\glslongtok}
5898 \glxtrparen{\protect\glabbrvfont{\the\glsshorttok}}%

```

5899 }

long-short-desc User supplies description. The long form is included in the name.

```
5900 \newabbreviationstyle{long-short-desc}%
5901 {%
5902   \renewcommand*{\CustomAbbreviationFields}{%
5903     name={\glxtrlongshortdescname},
5904     sort={\glxtrlongshortdescsort},%
5905     first={\protect\glsfirstlongfont{\the\glslongtok}%
5906       \protect\glxtrfullsep{\the\glslabeltok}%
5907       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
5908     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
5909       \protect\glxtrfullsep{\the\glslabeltok}%
5910       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
```

The text key should only have the short form.

```
5911   text={\protect\glsabbrvfont{\the\glsshorttok}},%
5912   plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
5913 }%
```

Unset the regular attribute if it has been set.

```
5914 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5915   \glshasattribute{\the\glslabeltok}{regular}%
5916   {%
5917     \glissetattribute{\the\glslabeltok}{regular}{false}%
5918   }%
5919   {}%
5920 }%
5921 }%
5922 {%
5923   \GlsXtrUseAbbrStyleFmts{long-short}%
5924 }
```

short-long Short form followed by long form in parenthesis on first use.

```
5925 \newabbreviationstyle{short-long}%
5926 {%
5927   \renewcommand*{\CustomAbbreviationFields}{%
5928     name={\protect\glsabbrvfont{\the\glsshorttok}},
5929     sort={\the\glsshorttok},
5930     description={\the\glslongtok},%
5931     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
5932       \protect\glxtrfullsep{\the\glslabeltok}%
5933       \glxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
5934     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
5935       \protect\glxtrfullsep{\the\glslabeltok}%
5936       \glxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
5937     plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
5938   }
```

Unset the regular attribute if it has been set.

```

5938 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5939 \glshasattribute{\the\glslabeltok}{regular}%
5940 {%
5941 \glsssetattribute{\the\glslabeltok}{regular}{false}%
5942 }%
5943 {}%
5944 }%
5945 }%
5946 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5947 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
5948 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvdefaultfont{##1}}%
5949 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvdefaultfont{##1}}%
5950 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
5951 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5952 \renewcommand*{\glsxtrfullformat}[2]{%
5953 \glssfirstabbrvfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5954 \ifglsxtrininsertinside\else##2\fi
5955 \glsxtrfullsep{##1}%
5956 \glsxtrparen{\glssfirstlongfont{\glssaccesslong{##1}}}%
5957 }%
5958 \renewcommand*{\glsxtrfullplformat}[2]{%
5959 \glssfirstabbrvfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5960 \ifglsxtrininsertinside\else##2\fi
5961 \glsxtrfullsep{##1}%
5962 \glsxtrparen{\glssfirstlongfont{\glssaccesslongpl{##1}}}%
5963 }%
5964 \renewcommand*{\Glsxtrfullformat}[2]{%
5965 \glssfirstabbrvfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
5966 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5967 \glsxtrparen{\glssfirstlongfont{\glssaccesslong{##1}}}%
5968 }%
5969 \renewcommand*{\Glsxtrfullplformat}[2]{%
5970 \glssfirstabbrvfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
5971 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
5972 \glsxtrparen{\glssfirstlongfont{\glssaccesslongpl{##1}}}%
5973 }%
5974 }

```

ortlongdescsort

```

5975 \newcommand*{\glsxtrshortlongdescsort}{\the\glssshorttok}

```

ortlongdescname

```

5976 \newcommand*{\glsxtrshortlongdescname}{%
5977 \protect\glssabbrvfont{\the\glssshorttok}
5978 \glsxtrparen{\protect\glsslongfont{\the\glsslongtok}}%

```


5979 }

short-long-desc User supplies description. The long form is included in the name.

```
5980 \newabbreviationstyle{short-long-desc}%
5981 {%
5982   \renewcommand*{\CustomAbbreviationFields}{%
5983     name={\glstrshortlongdescname},
5984     sort={\glstrshortlongdescsort},
5985     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
5986       \protect\glstrfullsep{\the\glslabeltok}%
5987       \glstrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
5988     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
5989       \protect\glstrfullsep{\the\glslabeltok}%
5990       \glstrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
5991     text={\protect\glsabbrvfont{\the\glsshorttok}},%
5992     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
5993   }%
```

Unset the regular attribute if it has been set.

```
5994   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5995     \glshasattribute{\the\glslabeltok}{regular}%
5996     {%
5997       \glissetattribute{\the\glslabeltok}{regular}{false}%
5998     }%
5999   }%
6000 }%
6001 }%
6002 {%
6003   \GlsXtrUseAbbrStyleFmts{short-long}%
6004 }
```

ongfootnotefont Only used by the “footnote” styles.

```
6005 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
6006 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%
```

xtrabbrvfootnote

`\glstrabbrvfootnote{<label>}{<long>}`

Command used by footnote abbreviation styles. The default definition ignores the first argument. The second argument *<long>* includes the font changing command and may be the singular or plural form, depending on the command that was used (for example, `\gls` or `\glspl`).

```
6007 \newcommand*{\glstrabbrvfootnote}[2]{\footnote{#2}}
```

footnote Short form followed by long form in footnote on first use.

```

6008 \newabbreviationstyle{footnote}%
6009 {%
6010   \renewcommand*{\CustomAbbreviationFields}{%
6011     name={\protect\glsabbrvfont{\the\glsshorttok}},
6012     sort={\the\glsshorttok},
6013     description={\the\glslongtok},%

6014     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6015       \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6016       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
6017     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6018       \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6019       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%

6020     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.

```

6021 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6022   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
6023   \glshasattribute{\the\glslabeltok}{regular}%
6024   {%
6025     \glssetattribute{\the\glslabeltok}{regular}{false}%
6026   }%
6027   {}%
6028 }%
6029 }%
6030 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6031 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6032 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6033 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6034 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6035 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

6036 \renewcommand*{\glsxtrfullformat}[2]{%
6037   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6038   \ifglsxtrininsertinside\else##2\fi
6039   \protect\glsxtrabbrvfootnote{##1}%
6040   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6041 }%
6042 \renewcommand*{\glsxtrfullplformat}[2]{%
6043   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6044   \ifglsxtrininsertinside\else##2\fi
6045   \protect\glsxtrabbrvfootnote{##1}%
6046   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6047 }%

```

```

6048 \renewcommand*{\Glsxtrfullformat}[2]{%
6049   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6050   \ifglxtrinsertinside\else##2\fi
6051   \protect\glxtrabbrvfootnote{##1}%
6052   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6053 }%
6054 \renewcommand*{\Glsxtrfullplformat}[2]{%
6055   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6056   \ifglxtrinsertinside\else##2\fi
6057   \protect\glxtrabbrvfootnote{##1}%
6058   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6059 }%

```

The first use full form and the inline full form use the short (long) style.

```

6060 \renewcommand*{\glxtrinlinefullformat}[2]{%
6061   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6062   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6063   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6064 }%
6065 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6066   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6067   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6068   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6069 }%
6070 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6071   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6072   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6073   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
6074 }%
6075 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6076   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6077   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6078   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6079 }%
6080 }

```

short-footnote

```
6081 \letabbreviationstyle{short-footnote}{footnote}
```

postfootnote Similar to the above but the footnote is placed afterwards, outside the link. This avoids nested links and can also move the footnote marker after any following punctuation mark. Pre v1.07 included `\footnote` in the first keys, which was incorrect as it caused duplicate footnotes.

```

6082 \newabbreviationstyle{postfootnote}%
6083 {%
6084   \renewcommand*{\CustomAbbreviationFields}{%
6085     name={\protect\glsabbrvfont{\the\glsshorttok}},
6086     sort={\the\glsshorttok},
6087     description={\the\glslongtok},%
6088     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
6089     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%

```

6090 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.

6091 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6092 \csdef{glsxtrpostlink\glscategorylabel}{%
6093 \glsxtrifwasfirstuse
6094 {%

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

6095 \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}%
6096 {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
6097 }%
6098 }%
6099 }%
6100 \glsattribute{\the\glslabeltok}{regular}%
6101 {%
6102 \glssetattribute{\the\glslabeltok}{regular}{false}%
6103 }%
6104 }%
6105 }%

The footnote needs to be suppressed in the inline form, so \glsxtrfull must set the first use switch off.

6106 \renewcommand*{\glsxtrsetupfulldefs}{%
6107 \let\glsxtrifwasfirstuse\@secondoftwo
6108 }%
6109 }%
6110 {%

In case the user wants to mix and match font styles, these are redefined here.

6111 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6112 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6113 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6114 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6115 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

The full format displays the short form. The long form is deferred.

6116 \renewcommand*{\glsxtrfullformat}[2]{%
6117 \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
6118 \ifglsxtrininsertinside\else##2\fi
6119 }%
6120 \renewcommand*{\glsxtrfullplformat}[2]{%
6121 \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
6122 \ifglsxtrininsertinside\else##2\fi
6123 }%
6124 \renewcommand*{\Glsxtrfullformat}[2]{%
6125 \glsfirstabbrvfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
6126 \ifglsxtrininsertinside\else##2\fi
6127 }%

```

6128 \renewcommand*{\Glsxtrfullplformat}[2]{%
6129   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6130   \ifglxtrinsertinside\else##2\fi
6131 }%

```

The first use full form and the inline full form use the short (long) style.

```

6132 \renewcommand*{\glxtrinlinefullformat}[2]{%
6133   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6134   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6135   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6136 }%
6137 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6138   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6139   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6140   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6141 }%
6142 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6143   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6144   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6145   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6146 }%
6147 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6148   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6149   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6150   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6151 }%
6152 }

```

rt-postfootnote

```

6153 \letabbreviationstyle{short-postfootnote}{postfootnote}

```

short Provide a style that only displays the short form on first use, but the short and long form can be displayed with the “full” commands that use the inline format. If the user supplies a description, the long form won’t be displayed in the predefined glossary styles, but the post description hook can be employed to automatically insert it.

```

6154 \newabbreviationstyle{short}%
6155 {%
6156   \renewcommand*{\CustomAbbreviationFields}{%
6157     name={\protect\glsabbrvfont{\the\glsshorttok}},
6158     sort={\the\glsshorttok},
6159     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
6160     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
6161     text={\protect\glsabbrvfont{\the\glsshorttok}},
6162     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
6163     description={\the\glslongtok}}%
6164   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6165     \glssetattribute{\the\glslabeltok}{regular}{true}}%
6166 }%
6167 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6168 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
6169 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
6170 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
6171 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
6172 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

6173 \renewcommand*\glxtrinlinefullformat[2]{%
6174   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
6175   \ifglxtrininsertinside##2\fi}%
6176 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6177 \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6178 }%
6179 \renewcommand*\glxtrinlinefullplformat[2]{%
6180   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
6181   \ifglxtrininsertinside##2\fi}%
6182 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6183 \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6184 }%
6185 \renewcommand*\Glsxtrinlinefullformat[2]{%
6186   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
6187   \ifglxtrininsertinside##2\fi}%
6188 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6189 \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
6190 }%
6191 \renewcommand*\Glsxtrinlinefullplformat[2]{%
6192   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
6193   \ifglxtrininsertinside##2\fi}%
6194 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6195 \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
6196 }%

```

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.

```

6197 \renewcommand*\glxtrfullformat[2]{%
6198   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
6199 \ifglxtrininsertinside\else##2\fi
6200 }%
6201 \renewcommand*\glxtrfullplformat[2]{%
6202   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
6203 \ifglxtrininsertinside\else##2\fi
6204 }%
6205 \renewcommand*\Glsxtrfullformat[2]{%
6206   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
6207 \ifglxtrininsertinside\else##2\fi
6208 }%
6209 \renewcommand*\Glsxtrfullplformat[2]{%
6210   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
6211 \ifglxtrininsertinside\else##2\fi

```

```
6212 }%
6213 }
```

Set this as the default style for acronyms:

```
6214 \setabbreviationstyle[acronym]{short}
```

short-nolong

```
6215 \letabbreviationstyle{short-nolong}{short}
```

rt-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```
6216 \newabbreviationstyle{short-nolong-noreg}%
6217 {%
6218   \GlsXtrUseAbbrStyleSetup{short-nolong}%
```

Unset the regular attribute if it has been set.

```
6219   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6220     \glshasattribute{\the\glslabeltok}{regular}%
6221     {%
6222       \glissetattribute{\the\glslabeltok}{regular}{false}%
6223     }%
6224   }%
6225 }%
6226 }%
6227 {%
6228   \GlsXtrUseAbbrStyleFmts{short-nolong}%
6229 }
```

trshortdescname

```
6230 \newcommand*{\glxtrshortdescname}{%
6231   \protect\glsabbrvfont{\the\glsshorttok}%
6232 }
```

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.

```
6233 \newabbreviationstyle{short-desc}%
6234 {%
6235   \renewcommand*{\CustomAbbreviationFields}{%
6236     name={\glxtrshortdescname},
6237     sort={\the\glsshorttok},
6238     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
6239     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
6240     text={\protect\glsabbrvfont{\the\glsshorttok}},
6241     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
6242     description={\the\glslongtok}}%
6243   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6244     \glissetattribute{\the\glslabeltok}{regular}{true}}%
6245 }%
6246 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6247 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
6248 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6249 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6250 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6251 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short format followed by the long form in parentheses.

```
6252 \renewcommand*{\glxtrinlinefullformat}[2]{%
6253   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6254   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6255   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6256 }%
6257 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6258   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6259   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6260   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6261 }%
6262 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6263   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6264   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6265   \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
6266 }%
6267 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6268   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6269   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6270   \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
6271 }%
```

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.

```
6272 \renewcommand*{\glxtrfullformat}[2]{%
6273   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6274   \ifglxtrininsertinside\else##2\fi
6275 }%
6276 \renewcommand*{\glxtrfullplformat}[2]{%
6277   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6278   \ifglxtrininsertinside\else##2\fi
6279 }%
6280 \renewcommand*{\Glsxtrfullformat}[2]{%
6281   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6282   \ifglxtrininsertinside\else##2\fi
6283 }%
6284 \renewcommand*{\Glsxtrfullplformat}[2]{%
6285   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6286   \ifglxtrininsertinside\else##2\fi
6287 }%
6288 }
```

ort-nolong-desc


```
6289 \letabbreviationstyle{short-nolong-desc}{short-desc}
```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```
6290 \newabbreviationstyle{short-nolong-desc-noreg}%
```

```
6291 {%
```

```
6292 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
```

Unset the regular attribute if it has been set.

```
6293 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
6294 \glshasattribute{\the\glslabeltok}{regular}%
```

```
6295 {%
```

```
6296 \glissetattribute{\the\glslabeltok}{regular}{false}%
```

```
6297 }%
```

```
6298 {}%
```

```
6299 }%
```

```
6300 }%
```

```
6301 {%
```

```
6302 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
```

```
6303 }
```

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.

```
6304 \newabbreviationstyle{long-desc}%
```

```
6305 {%
```

```
6306 \renewcommand*{\CustomAbbreviationFields}{%
```

```
6307 name={\protect\protect\glslongfont{\the\glslongtok}},
```

```
6308 sort={\the\glslongtok},
```

```
6309 first={\protect\glslongfont{\the\glslongtok}},
```

```
6310 firstplural={\protect\glslongfont{\the\glslongpltok}},
```

```
6311 text={\glslongfont{\the\glslongtok}},
```

```
6312 plural={\glslongfont{\the\glslongpltok}}%
```

```
6313 }%
```

```
6314 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
6315 \glissetattribute{\the\glslabeltok}{regular}{true}}%
```

```
6316 }%
```

```
6317 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6318 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
```

```
6319 \renewcommand*{\glssabrvfont}[1]{\glssabrvdefaultfont{##1}}%
```

```
6320 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
```

```
6321 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
```

```
6322 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
6323 \renewcommand*{\glxtrsubsequentfmt}[2]{%
```

```
6324 \glslongfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
```

```
6325 \ifglxtrininsertinside \else##2\fi
```

```
6326 }%
```

```

6327 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
6328   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
6329   \ifglxtrinsertinside \else##2\fi
6330 }%
6331 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
6332   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
6333   \ifglxtrinsertinside \else##2\fi
6334 }%
6335 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
6336   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
6337   \ifglxtrinsertinside \else##2\fi
6338 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

6339 \renewcommand*{\glxtrinlinefullformat}[2]{%
6340   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6341   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6342   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6343 }%
6344 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6345   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6346   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6347   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6348 }%
6349 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6350   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6351   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6352   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6353 }%
6354 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6355   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6356   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6357   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6358 }%

```

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.

```

6359 \renewcommand*{\glxtrfullformat}[2]{%
6360   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6361   \ifglxtrinsertinside\else##2\fi
6362 }%
6363 \renewcommand*{\glxtrfullplformat}[2]{%
6364   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6365   \ifglxtrinsertinside\else##2\fi
6366 }%
6367 \renewcommand*{\Glsxtrfullformat}[2]{%
6368   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6369   \ifglxtrinsertinside\else##2\fi
6370 }%
6371 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

6372 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6373 \ifglxtrinsertinside\else##2\fi
6374 }%
6375 }

```

ng-noshort-desc Provide a synonym that matches similar styles.

```
6376 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

short-desc-noreg Like long-noshort-desc but doesn't set the regular attribute.

```

6377 \newabbreviationstyle{long-noshort-desc-noreg}%
6378 {%
6379 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%

Unset the regular attribute if it has been set.

6380 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6381 \glsattribute{\the\glslabeltok}{regular}%
6382 {%
6383 \glssetattribute{\the\glslabeltok}{regular}{false}%
6384 }%
6385 {}%
6386 }%
6387 }%
6388 {%
6389 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
6390 }

```

long It doesn't really make a great deal of sense to have a long-only style that doesn't have a description (unless no glossary is required), but the best course of action here is to use the short form as the name and the long form as the description.

```

6391 \newabbreviationstyle{long}%
6392 {%
6393 \renewcommand*{\CustomAbbreviationFields}{%
6394 name={\protect\glsabbrvfont{\the\glsshorttok}},
6395 sort={\the\glsshorttok},
6396 first={\protect\glsfirstlongfont{\the\glslongtok}},
6397 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
6398 text={\glslongfont{\the\glslongtok}},
6399 plural={\glslongfont{\the\glslongpltok}},%
6400 description={\the\glslongtok}%
6401 }%
6402 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6403 \glssetattribute{\the\glslabeltok}{regular}{true}}%
6404 }%
6405 {%
6406 \GlsXtrUseAbbrStyleFmts{long-desc}%
6407 }

```

long-noshort Provide a synonym that matches similar styles.

```
6408 \letabbreviationstyle{long-noshort}{long}
```

g-noshort-noreg Like long-noshort but doesn't set the regular attribute.

```
6409 \newabbreviationstyle{long-noshort-noreg}%
6410 {%
6411   \GlsXtrUseAbbrStyleSetup{long-noshort}%
6412   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6413     \glshasattribute{\the\glslabeltok}{regular}%
6414     {%
6415       \glissetattribute{\the\glslabeltok}{regular}{false}%
6416     }%
6417   }%
6418 }%
6419 }%
6420 {%
6421   \GlsXtrUseAbbrStyleFmts{long-noshort}%
6422 }
```

1.6.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

`\glxtrscfont` Maintained for backward-compatibility.

```
6423 \newcommand*{\glxtrscfont}[1]{\textsc{#1}}
```

`\glsabbrvscfont` Added for consistent naming.

```
6424 \newcommand*{\glsabbrvscfont}{\glxtrscfont}
```

`sxtrfirstscfont` Maintained for backward-compatibility.

```
6425 \newcommand*{\sxtrfirstscfont}[1]{\glsabbrvscfont{#1}}
```

`irstabbrvscfont` Added for consistent naming.

```
6426 \newcommand*{\glfirstabbrvscfont}{\glxtrfirstscfont}
```

and for the default short form suffix:

`\glxtrscsuffix`

```
6427 \newcommand*{\glxtrscsuffix}{\glstextup{\glxtrabbrvpluralsuffix}}
```

`long-short-sc`

```
6428 \newabbreviationstyle{long-short-sc}%
6429 {%
6430   \renewcommand*{\CustomAbbreviationFields}{%
6431     name={\protect\glsabbrvscfont{\the\glsshorttok}},
6432     sort={\the\glsshorttok},
6433     first={\protect\glfirstlongdefaultfont{\the\glslongtok}%
6434       \protect\glxtrfullsep{\the\glslabeltok}%
6435       \glxtrparen{\protect\glfirstabbrvscfont{\the\glsshorttok}}},%
6436     firstplural={\protect\glfirstlongdefaultfont{\the\glslongpltok}%
6437       \protect\glxtrfullsep{\the\glslabeltok}%
6438       \glxtrparen{\protect\glfirstabbrvscfont{\the\glsshorttok}}},%
6439   }
```

```

6437 \protect\glxtrfullsep{\the\glslabeltok}%
6438 \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
6439 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}},%
6440 description={\the\glslongtok}}%
6441 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6442 \glsattribute{\the\glslabeltok}{regular}%
6443 {%
6444 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
6445 }%
6446 {}%
6447 }%
6448 }%
6449 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

6450 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
6451 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
6452 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%

```

Use the default long fonts.

```

6453 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6454 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6455 \renewcommand*{\glxtrfullformat}[2]{%
6456 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6457 \ifglxtrininsertinside\else##2\fi
6458 \glxtrfullsep{##1}%
6459 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
6460 }%
6461 \renewcommand*{\glxtrfullplformat}[2]{%
6462 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6463 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6464 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
6465 }%
6466 \renewcommand*{\Glsxtrfullformat}[2]{%
6467 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6468 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6469 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
6470 }%
6471 \renewcommand*{\Glsxtrfullplformat}[2]{%
6472 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6473 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6474 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
6475 }%
6476 }

```

g-short-sc-desc

```

6477 \newabbreviationstyle{long-short-sc-desc}%
6478 {%

```

```

6479 \renewcommand*{\CustomAbbreviationFields}{%
6480   name={\glxtrlongshortdescname},
6481   sort={\glxtrlongshortdescsort},%
6482   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
6483     \protect\glxtrfullsep{\the\glslabeltok}%
6484     \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
6485   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
6486     \protect\glxtrfullsep{\the\glslabeltok}%
6487     \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
6488   text={\protect\glssabbrvscfont{\the\glsshorttok}},%
6489   plural={\protect\glssabbrvscfont{\the\glsshortpltok}}}%
6490 }%

```

Unset the regular attribute if it has been set.

```

6491 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6492   \glshasattribute{\the\glslabeltok}{regular}%
6493   {%
6494     \glissetattribute{\the\glslabeltok}{regular}{false}%
6495   }%
6496   {}%
6497 }%
6498 }%
6499 {%

```

As long-short-sc style:

```

6500 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
6501 }

```

Now the short (long) version

```

6502 \newabbreviationstyle{short-sc-long}%
6503 {%
6504   \renewcommand*{\CustomAbbreviationFields}{%
6505     name={\protect\glssabbrvscfont{\the\glsshorttok}},
6506     sort={\the\glsshorttok},
6507     description={\the\glslongtok},%
6508     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
6509       \protect\glxtrfullsep{\the\glslabeltok}%
6510       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
6511     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
6512       \protect\glxtrfullsep{\the\glslabeltok}%
6513       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
6514     plural={\protect\glssabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

6515 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6516   \glshasattribute{\the\glslabeltok}{regular}%
6517   {%
6518     \glissetattribute{\the\glslabeltok}{regular}{false}%
6519   }%
6520   {}%
6521 }%

```

6522 }%

6523 {%

Use smallcaps and adjust the plural suffix to revert to upright.

```
6524 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
6525 \renewcommand*{\glabbrvfont}[1]{\glabbrvscfont{##1}}%
6526 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvscfont{##1}}%
6527 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
6528 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
6529 \renewcommand*{\glxtrfullformat}[2]{%
6530   \glfirstabbrvscfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6531   \ifglxtrininsertinside\else##2\fi
6532   \glxtrfullsep{##1}%
6533   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
6534 }%
6535 \renewcommand*{\glxtrfullplformat}[2]{%
6536   \glfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6537   \ifglxtrininsertinside\else##2\fi
6538   \glxtrfullsep{##1}%
6539   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
6540 }%
6541 \renewcommand*{\Glsxtrfullformat}[2]{%
6542   \glfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6543   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6544   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
6545 }%
6546 \renewcommand*{\Glsxtrfullplformat}[2]{%
6547   \glfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6548   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6549   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
6550 }%
6551 }
```

As before but user provides description

```
6552 \newabbreviationstyle{short-sc-long-desc}%
6553 {%
6554   \renewcommand*{\CustomAbbreviationFields}{%
6555     name={\glxtrshortlongdescname},
6556     sort={\glxtrshortlongdescsort},
6557     first={\protect\glfirstabbrvscfont{\the\glsshorttok}}%
6558     \protect\glxtrfullsep{\the\glslabeltok}%
6559     \glxtrparen{\protect\glfirstlongdefaultfont{\the\glslongtok}}},%
6560     firstplural={\protect\glfirstabbrvscfont{\the\glsshortpltok}}%
6561     \protect\glxtrfullsep{\the\glslabeltok}%
6562     \glxtrparen{\protect\glfirstlongdefaultfont{\the\glslongpltok}}},%
6563     text={\protect\glabbrvscfont{\the\glsshorttok}},%
6564     plural={\protect\glabbrvscfont{\the\glsshortpltok}}%
6565   }%
```

Unset the regular attribute if it has been set.

```
6566 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6567   \glshasattribute{\the\glslabeltok}{regular}%
6568   {%
6569     \glissetattribute{\the\glslabeltok}{regular}{false}%
6570   }%
6571   {}%
6572 }%
6573 }%
6574 {%
```

As short-sc-long style:

```
6575 \GlsXtrUseAbbrStyleFmts{short-sc-long}%
6576 }
```

short-sc

```
6577 \newabbreviationstyle{short-sc}%
6578 {%
6579   \renewcommand*{\CustomAbbreviationFields}{%
6580     name={\protect\glsabbrvscfont{\the\glsshorttok}},
6581     sort={\the\glsshorttok},
6582     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
6583     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
6584     text={\protect\glsabbrvscfont{\the\glsshorttok}},
6585     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
6586     description={\the\glslongtok}}%
6587   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6588     \glissetattribute{\the\glslabeltok}{regular}{true}}%
6589 }%
6590 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
6591 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
6592 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
6593 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
6594 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6595 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
6596 \renewcommand*{\glxtrinlinefullformat}[2]{%
6597   \protect\glsfirstabbrvscfont{\glsaccessshort{##1}}%
6598   \ifglxtrininsertinside##2\fi}%
6599   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6600   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
6601 }%
6602 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6603   \protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}%
6604   \ifglxtrininsertinside##2\fi}%
6605   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6606   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%

```



```

6607 }%
6608 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6609   \protect\glsfirstabbrvscfont{\glsaccessshort{##1}%
6610     \ifglxtrinsertinside##2\fi}%
6611   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6612   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
6613 }%
6614 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6615   \protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}%
6616     \ifglxtrinsertinside##2\fi}%
6617   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6618   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
6619 }%

```

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.

```

6620 \renewcommand*{\glxtrfullformat}[2]{%
6621   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6622   \ifglxtrinsertinside\else##2\fi
6623 }%
6624 \renewcommand*{\glxtrfullplformat}[2]{%
6625   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6626   \ifglxtrinsertinside\else##2\fi
6627 }%
6628 \renewcommand*{\Glsxtrfullformat}[2]{%
6629   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6630   \ifglxtrinsertinside\else##2\fi
6631 }%
6632 \renewcommand*{\Glsxtrfullplformat}[2]{%
6633   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6634   \ifglxtrinsertinside\else##2\fi
6635 }%
6636 }

```

short-sc-nolong

```

6637 \letabbreviationstyle{short-sc-nolong}{short-sc}

```

short-sc-desc

```

6638 \newabbreviationstyle{short-sc-desc}%
6639 {%
6640   \renewcommand*{\CustomAbbreviationFields}{%
6641     name={\glxtrshortdescname},
6642     sort={\the\glsshorttok},
6643     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
6644     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
6645     text={\protect\glsabbrvscfont{\the\glsshorttok}},
6646     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
6647     description={\the\glslongtok}}%
6648 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

6649 \glssetattribute{\the\glslabeltok}{regular}{true}}%
6650 }%
6651 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

6652 \renewcommand*\abbrvpluralsuffix{\protect\glstrscsuffix}%
6653 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
6654 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
6655 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
6656 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

6657 \renewcommand*\glsxtrinlinefullformat[2]{%
6658   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6659   \ifglsxtrininsertinside\else##2\fi\glstrfullsep{##1}%
6660   \glstrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
6661 }%
6662 \renewcommand*\glsxtrinlinefullplformat[2]{%
6663   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6664   \ifglsxtrininsertinside\else##2\fi\glstrfullsep{##1}%
6665   \glstrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
6666 }%
6667 \renewcommand*\Glsxtrinlinefullformat[2]{%
6668   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6669   \ifglsxtrininsertinside\else##2\fi\glstrfullsep{##1}%
6670   \glstrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
6671 }%
6672 \renewcommand*\Glsxtrinlinefullplformat[2]{%
6673   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6674   \ifglsxtrininsertinside\else##2\fi\glstrfullsep{##1}%
6675   \glstrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
6676 }%

```

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.

```

6677 \renewcommand*\glstrfullformat[2]{%
6678   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6679   \ifglsxtrininsertinside\else##2\fi
6680 }%
6681 \renewcommand*\glstrfullplformat[2]{%
6682   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6683   \ifglsxtrininsertinside\else##2\fi
6684 }%
6685 \renewcommand*\Glsxtrfullformat[2]{%
6686   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6687   \ifglsxtrininsertinside\else##2\fi
6688 }%
6689 \renewcommand*\Glsxtrfullplformat[2]{%
6690   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6691   \ifglsxtrininsertinside\else##2\fi
6692 }%

```

6693 }

-sc-nolong-desc

6694 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

long-noshort-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like \glxtrshort.

6695 \newabbreviationstyle{long-noshort-sc}{%

6696 {%

6697 \renewcommand*{\CustomAbbreviationFields}{%

6698 name={\protect\glsabbrvscfont{\the\glsshorttok}},

6699 sort={\the\glsshorttok},

6700 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},

6701 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},

6702 text={\protect\glslongdefaultfont{\the\glslongtok}},

6703 plural={\protect\glslongdefaultfont{\the\glslongpltok}},%

6704 description={\the\glslongtok}%

6705 }%

6706 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

6707 \glssetattribute{\the\glslabeltok}{regular}{true}}%

6708 }%

6709 {%

Use smallcaps and adjust the plural suffix to revert to upright.

6710 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%

6711 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%

6712 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%

6713 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%

6714 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

The format for subsequent use (not used when the regular attribute is set).

6715 \renewcommand*{\glxtrsubsequentfmt}[2]{%

6716 \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%

6717 \ifglxtrininsertinside \else##2\fi

6718 }%

6719 \renewcommand*{\glxtrsubsequentplfmt}[2]{%

6720 \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%

6721 \ifglxtrininsertinside \else##2\fi

6722 }%

6723 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%

6724 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%

6725 \ifglxtrininsertinside \else##2\fi

6726 }%

6727 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%

6728 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%

6729 \ifglxtrininsertinside \else##2\fi

6730 }%

The inline full form displays the long format followed by the short form in parentheses.

6731 \renewcommand*{\glxtrininlinefullformat}[2]{%

```

6732 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6733 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6734 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
6735 }%
6736 \renewcommand*{\glxtrinelinefullplformat}[2]{%
6737 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6738 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6739 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
6740 }%
6741 \renewcommand*{\Glsxtrinelinefullformat}[2]{%
6742 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6743 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6744 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
6745 }%
6746 \renewcommand*{\Glsxtrinelinefullplformat}[2]{%
6747 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6748 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6749 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
6750 }%

```

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.

```

6751 \renewcommand*{\glxtrfullformat}[2]{%
6752 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6753 \ifglxtrinsertinside\else##2\fi
6754 }%
6755 \renewcommand*{\glxtrfullplformat}[2]{%
6756 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6757 \ifglxtrinsertinside\else##2\fi
6758 }%
6759 \renewcommand*{\Glsxtrfullformat}[2]{%
6760 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6761 \ifglxtrinsertinside\else##2\fi
6762 }%
6763 \renewcommand*{\Glsxtrfullplformat}[2]{%
6764 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6765 \ifglxtrinsertinside\else##2\fi
6766 }%
6767 }

```

long-sc Backward compatibility:

```

6768 \@glxtr@deprecated@abbrstyle{long-sc}{long-noshort-sc}

```

noshort-sc-desc The smallcaps font will only be used if the short form is explicitly invoked through commands like \glsshort.

```

6769 \newabbreviationstyle{long-noshort-sc-desc}%
6770 {%
6771 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
6772 }%
6773 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```
6774 \renewcommand*\abbrvpluralsuffix{\protect\glxtrscsuffix}%
6775 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
6776 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
6777 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
6778 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
6779 \renewcommand*\glxtrsubsequentfmt}[2]{%
6780   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
6781   \ifglxtrininsertinside \else##2\fi
6782 }%
6783 \renewcommand*\glxtrsubsequentplfmt}[2]{%
6784   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
6785   \ifglxtrininsertinside \else##2\fi
6786 }%
6787 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
6788   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
6789   \ifglxtrininsertinside \else##2\fi
6790 }%
6791 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
6792   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
6793   \ifglxtrininsertinside \else##2\fi
6794 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
6795 \renewcommand*\glxtrinlinefullformat}[2]{%
6796   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6797   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6798   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
6799 }%
6800 \renewcommand*\glxtrinlinefullplformat}[2]{%
6801   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6802   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6803   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
6804 }%
6805 \renewcommand*\Glsxtrinlinefullformat}[2]{%
6806   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6807   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6808   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
6809 }%
6810 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
6811   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6812   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6813   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
6814 }%
```

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.

```
6815 \renewcommand*\glxtrfullformat}[2]{%
```

```

6816 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6817 \ifglxtrinsertinside\else##2\fi
6818 }%
6819 \renewcommand*{\glxtrfullplformat}[2]{%
6820 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6821 \ifglxtrinsertinside\else##2\fi
6822 }%
6823 \renewcommand*{\Glsxtrfullformat}[2]{%
6824 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6825 \ifglxtrinsertinside\else##2\fi
6826 }%
6827 \renewcommand*{\Glsxtrfullplformat}[2]{%
6828 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6829 \ifglxtrinsertinside\else##2\fi
6830 }%
6831 }

```

long-desc-sc Backward compatibility:

```

6832 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

6833 \newabbreviationstyle{short-sc-footnote}%
6834 {%
6835 \renewcommand*{\CustomAbbreviationFields}{%
6836 name={\protect\glsabbrvscfont{\the\glsshorttok}},
6837 sort={\the\glsshorttok},
6838 description={\the\glslongtok},%
6839 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%
6840 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
6841 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
6842 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}}%
6843 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
6844 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
6845 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

6846 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6847 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
6848 \glshasattribute{\the\glslabeltok}{regular}%
6849 {%
6850 \glssetattribute{\the\glslabeltok}{regular}{false}%
6851 }%
6852 {}%
6853 }%
6854 }%
6855 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

6856 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%

```

```

6857 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
6858 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
6859 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
6860 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

6861 \renewcommand*\glsxtrfullformat[2]{%
6862   \glsfirstabbrvscfont{\glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
6863   \ifglsxtrininsertinside\else##2\fi
6864   \protect\glsxtrabbrvfootnote{##1}%
6865   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6866 }%
6867 \renewcommand*\glsxtrfullplformat[2]{%
6868   \glsfirstabbrvscfont{\glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6869   \ifglsxtrininsertinside\else##2\fi
6870   \protect\glsxtrabbrvfootnote{##1}%
6871   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6872 }%
6873 \renewcommand*\Glsxtrfullformat[2]{%
6874   \glsfirstabbrvscfont{\Glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
6875   \ifglsxtrininsertinside\else##2\fi
6876   \protect\glsxtrabbrvfootnote{##1}%
6877   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6878 }%
6879 \renewcommand*\Glsxtrfullplformat[2]{%
6880   \glsfirstabbrvscfont{\Glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6881   \ifglsxtrininsertinside\else##2\fi
6882   \protect\glsxtrabbrvfootnote{##1}%
6883   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6884 }%

```

The first use full form and the inline full form use the short (long) style.

```

6885 \renewcommand*\glsxtrinlinefullformat[2]{%
6886   \glsfirstabbrvscfont{\glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
6887   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6888   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6889 }%
6890 \renewcommand*\glsxtrinlinefullplformat[2]{%
6891   \glsfirstabbrvscfont{\glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6892   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6893   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6894 }%
6895 \renewcommand*\Glsxtrinlinefullformat[2]{%
6896   \glsfirstabbrvscfont{\Glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
6897   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6898   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6899 }%
6900 \renewcommand*\Glsxtrinlinefullplformat[2]{%
6901   \glsfirstabbrvscfont{\Glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6902   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%

```

```

6903 \glxtrparen{\glsfirstlongfootnotefont{\glaccesslongpl{##1}}}%
6904 }%
6905 }

```

footnote-sc Backward compatibility:

```

6906 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}

```

sc-postfootnote

```

6907 \newabbreviationstyle{short-sc-postfootnote}%
6908 {%
6909 \renewcommand*{\CustomAbbreviationFields}{%
6910 name={\protect\glabbrvscfont{\the\glsshorttok}},
6911 sort={\the\glsshorttok},
6912 description={\the\glslongtok},%
6913 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
6914 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
6915 plural={\protect\glabbrvscfont{\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.

```

6916 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6917 \csdef{glxtrpostlink\glscategorylabel}{%
6918 \glxtrifwasfirstuse
6919 }%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

6920 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
6921 {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
6922 }%
6923 {}%
6924 }%
6925 \glshasattribute{\the\glslabeltok}{regular}%
6926 {%
6927 \glissetattribute{\the\glslabeltok}{regular}{false}%
6928 }%
6929 {}%
6930 }%

```

The footnote needs to be suppressed in the inline form, so \glxtrfull must set the first use switch off.

```

6931 \renewcommand*{\glxtrsetupfulldefs}{%
6932 \let\glxtrifwasfirstuse\@secondoftwo
6933 }%
6934 }%
6935 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

6936 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
6937 \renewcommand*{\glabbrvfont[1]{\glabbrvscfont{##1}}}%

```



```

6938 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
6939 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
6940 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

6941 \renewcommand*\glsxtrfullformat}[2]{%
6942   \glsfirstabbrvscfont{\glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
6943   \ifglsxtrininsertinside\else##2\fi
6944 }%
6945 \renewcommand*\glsxtrfullplformat}[2]{%
6946   \glsfirstabbrvscfont{\glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6947   \ifglsxtrininsertinside\else##2\fi
6948 }%
6949 \renewcommand*\Glsxtrfullformat}[2]{%
6950   \glsfirstabbrvscfont{\Glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
6951   \ifglsxtrininsertinside\else##2\fi
6952 }%
6953 \renewcommand*\Glsxtrfullplformat}[2]{%
6954   \glsfirstabbrvscfont{\Glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6955   \ifglsxtrininsertinside\else##2\fi
6956 }%

```

The first use full form and the inline full form use the short (long) style.

```

6957 \renewcommand*\glsxtrinlinefullformat}[2]{%
6958   \glsfirstabbrvscfont{\glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
6959   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6960   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6961 }%
6962 \renewcommand*\glsxtrinlinefullplformat}[2]{%
6963   \glsfirstabbrvscfont{\glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6964   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6965   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6966 }%
6967 \renewcommand*\Glsxtrinlinefullformat}[2]{%
6968   \glsfirstabbrvscfont{\Glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
6969   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6970   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6971 }%
6972 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
6973   \glsfirstabbrvscfont{\Glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6974   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6975   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6976 }%
6977 }

```

postfootnote-sc Backward compatibility:

```

6978 \@glsxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}

```

1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the `relsize` package, which must be loaded by the user. These styles all use:

```
\glxtrsmfont    Maintained for backward compatibility.
6979 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}

\glabbrvsmfont  Added for consistent naming.
6980 \newcommand*{\glabbrvsmfont}{\glxtrsmfont}

sxtrfirstsmfont Maintained for backward compatibility.
6981 \newcommand*{\glxtrfirstsmfont}[1]{\glabbrvsmfont{#1}}

irstabbrvsmfont Added for consistent naming.
6982 \newcommand*{\glfirstabbrvsmfont}{\glxtrfirstsmfont}

and for the default short form suffix:

\glxtrsmsuffix
6983 \newcommand*{\glxtrsmsuffix}{\glxtrabbrvpluralsuffix}

long-short-sm
6984 \newabbreviationstyle{long-short-sm}%
6985 {%
6986   \renewcommand*{\CustomAbbreviationFields}{%
6987     name={\protect\glabbrvsmfont{\the\glsshorttok}},
6988     sort={\the\glsshorttok},
6989     first={\protect\glfirstlongdefaultfont{\the\glslongtok}%
6990       \protect\glxtrfullsep{\the\glslabeltok}%
6991       \glxtrparen{\protect\glfirstabbrvsmfont{\the\glsshorttok}}},%
6992     firstplural={\protect\glfirstlongdefaultfont{\the\glslongpltok}%
6993       \protect\glxtrfullsep{\the\glslabeltok}%
6994       \glxtrparen{\protect\glfirstabbrvsmfont{\the\glsshortpltok}}},%
6995     plural={\protect\glabbrvsmfont{\the\glsshortpltok}},%
6996     description={\the\glslongtok}}%
6997   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6998     \glshasattribute{\the\glslabeltok}{regular}%
6999     {%
7000       \glissetattribute{\the\glslabeltok}{regular}{false}%
7001     }%
7002   }%
7003 }%
7004 }%
7005 {%
7006   \renewcommand*{\glabbrvfont}[1]{\glabbrvsmfont{##1}}%
7007   \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvsmfont{##1}}%
7008   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
```

Use the default long fonts.

```
7009 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7010 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
7011 \renewcommand*{\glsxtrfullformat}[2]{%
7012   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7013   \ifglsxtrinsertinside\else##2\fi
7014   \glsxtrfullsep{##1}%
7015   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7016 }%
7017 \renewcommand*{\glsxtrfullplformat}[2]{%
7018   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7019   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7020   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7021 }%
7022 \renewcommand*{\Glsxtrfullformat}[2]{%
7023   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7024   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7025   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7026 }%
7027 \renewcommand*{\Glsxtrfullplformat}[2]{%
7028   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7029   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7030   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7031 }%
7032 }
```

g-short-sm-desc

```
7033 \newabbreviationstyle{long-short-sm-desc}%
7034 {%
7035   \renewcommand*{\CustomAbbreviationFields}{%
7036     name={\glsxtrlongshortdescname},
7037     sort={\glsxtrlongshortdescsort},%
7038     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7039       \protect\glsxtrfullsep{\the\glslabeltok}%
7040       \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7041     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7042       \protect\glsxtrfullsep{\the\glslabeltok}%
7043       \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7044     text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
7045     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}%
7046   }%
```

Unset the regular attribute if it has been set.

```
7047 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7048   \glshasattribute{\the\glslabeltok}{regular}%
7049   {%
7050     \glssetattribute{\the\glslabeltok}{regular}{false}%
7051   }%
```

```

7052    {}%
7053  }%
7054 }%
7055 {%
    As long-short-sm style:
7056   \GlsXtrUseAbbrStyleFmts{long-short-sm}%
7057 }

```

short-sm-long Now the short (long) version

```

7058 \newabbreviationstyle{short-sm-long}%
7059 {%
7060   \renewcommand*{\CustomAbbreviationFields}{%
7061     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7062     sort={\the\glsshorttok},
7063     description={\the\glslongtok},%
7064     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
7065       \protect\glsxtrfullsep{\the\glslabeltok}%
7066       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7067     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
7068       \protect\glsxtrfullsep{\the\glslabeltok}%
7069       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7070     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7071   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7072     \glsattribute{\the\glslabeltok}{regular}%
7073     {%
7074       \glsattribute{\the\glslabeltok}{regular}{false}%
7075     }%
7076     {}%
7077   }%
7078 }%
7079 {%
7080   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
7081   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
7082   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsmsuffix}%
7083   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7084   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7085   \renewcommand*{\glsxtrfullformat}[2]{%
7086     \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7087     \ifglsxtrininsertinside\else##2\fi
7088     \glsxtrfullsep{##1}%
7089     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7090   }%
7091   \renewcommand*{\glsxtrfullplformat}[2]{%
7092     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7093     \ifglsxtrininsertinside\else##2\fi

```

```

7094 \glxtrfullsep{##1}%
7095 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7096 }%
7097 \renewcommand*{\Glsxtrfullformat}[2]{%
7098 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7099 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7100 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7101 }%
7102 \renewcommand*{\Glsxtrfullplformat}[2]{%
7103 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7104 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7105 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7106 }%
7107 }

```

rt-sm-long-desc As before but user provides description

```

7108 \newabbreviationstyle{short-sm-long-desc}%
7109 {%
7110 \renewcommand*{\CustomAbbreviationFields}{%
7111 name={\glxtrshortlongdescname},
7112 sort={\glxtrshortlongdescsort},
7113 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
7114 \protect\glxtrfullsep{\the\glslabeltok}%
7115 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7116 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
7117 \protect\glxtrfullsep{\the\glslabeltok}%
7118 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7119 text={\protect\glssabbrvsmfont{\the\glsshorttok}},%
7120 plural={\protect\glssabbrvsmfont{\the\glsshortpltok}}}%
7121 }%

```

Unset the regular attribute if it has been set.

```

7122 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7123 \glshasattribute{\the\glslabeltok}{regular}%
7124 {%
7125 \glissetattribute{\the\glslabeltok}{regular}{false}%
7126 }%
7127 {}%
7128 }%
7129 }%
7130 {%

```

As short-sm-long style:

```

7131 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
7132 }

```

short-sm

```

7133 \newabbreviationstyle{short-sm}%
7134 {%
7135 \renewcommand*{\CustomAbbreviationFields}{%

```

```

7136   name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7137   sort={\the\glsshorttok},
7138   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
7139   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
7140   text={\protect\glsabbrvsmfont{\the\glsshorttok}},
7141   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
7142   description={\the\glslongtok}}%
7143 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7144   \glssetattribute{\the\glslabeltok}{regular}{true}}%
7145 }%
7146 {%
7147 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7148 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7149 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%
7150 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7151 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7152 \renewcommand*\glsxtrinlinefullformat[2]{%
7153   \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
7154   \ifglsxtrininsertinside##2\fi}%
7155 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7156 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7157 }%
7158 \renewcommand*\glsxtrinlinefullplformat[2]{%
7159   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}%
7160   \ifglsxtrininsertinside##2\fi}%
7161 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7162 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7163 }%
7164 \renewcommand*\Glsxtrinlinefullformat[2]{%
7165   \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
7166   \ifglsxtrininsertinside##2\fi}%
7167 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7168 \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7169 }%
7170 \renewcommand*\Glsxtrinlinefullplformat[2]{%
7171   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}%
7172   \ifglsxtrininsertinside##2\fi}%
7173 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7174 \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7175 }%

```

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.

```

7176 \renewcommand*\glsxtrfullformat[2]{%
7177   \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7178   \ifglsxtrininsertinside\else##2\fi
7179 }%
7180 \renewcommand*\glsxtrfullplformat[2]{%

```

```

7181 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7182 \ifglxtrinsertinside\else##2\fi
7183 }%
7184 \renewcommand*{\Glsxtrfullformat}[2]{%
7185 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7186 \ifglxtrinsertinside\else##2\fi
7187 }%
7188 \renewcommand*{\Glsxtrfullplformat}[2]{%
7189 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7190 \ifglxtrinsertinside\else##2\fi
7191 }%
7192 }

```

short-sm-nolong

```

7193 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

7194 \newabbreviationstyle{short-sm-desc}%
7195 {%
7196 \renewcommand*{\CustomAbbreviationFields}{%
7197 name={\glsxtrshortdescname},
7198 sort={\the\glsshorttok},
7199 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
7200 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
7201 text={\protect\glsabbrvsmfont{\the\glsshorttok}},
7202 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
7203 description={\the\glslongtok}}%
7204 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7205 \glssetattribute{\the\glslabeltok}{regular}{true}}%
7206 }%
7207 {%
7208 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7209 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7210 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrrmsuffix}%
7211 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7212 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7213 \renewcommand*{\glsxtrinlinefullformat}[2]{%
7214 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7215 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7216 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7217 }%
7218 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7219 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7220 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7221 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7222 }%
7223 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7224 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%

```

```

7225 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7226 \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
7227 }%
7228 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7229 \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7230 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7231 \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
7232 }%

```

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.

```

7233 \renewcommand*{\glxtrfullformat}[2]{%
7234 \glfirstabbrvsmfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7235 \ifglxtrinsertinside\else##2\fi
7236 }%
7237 \renewcommand*{\glxtrfullplformat}[2]{%
7238 \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7239 \ifglxtrinsertinside\else##2\fi
7240 }%
7241 \renewcommand*{\Glsxtrfullformat}[2]{%
7242 \glfirstabbrvsmfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7243 \ifglxtrinsertinside\else##2\fi
7244 }%
7245 \renewcommand*{\Glsxtrfullplformat}[2]{%
7246 \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7247 \ifglxtrinsertinside\else##2\fi
7248 }%
7249 }

```

-sm-nolong-desc

```
7250 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}
```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

7251 \newabbreviationstyle{long-noshort-sm}%
7252 {%
7253 \renewcommand*{\CustomAbbreviationFields}{%
7254 name={\protect\glabbrvsmfont{\the\glsshorttok}},
7255 sort={\the\glsshorttok},
7256 first={\protect\glfirstlongdefaultfont{\the\glslongtok}},
7257 firstplural={\protect\glfirstlongdefaultfont{\the\glslongpltok}},
7258 text={\protect\glslongdefaultfont{\the\glslongtok}},
7259 plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7260 description={\the\glslongtok}%
7261 }%
7262 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7263 \glsssetattribute{\the\glslabeltok}{regular}{true}}%
7264 }%
7265 {%

```



```

7266 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7267 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7268 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrsmsuffix}%
7269 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7270 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7271 \renewcommand*\glsxtrsubsequentfmt}[2]{%
7272   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7273   \ifglsxtrininsertinside \else##2\fi
7274 }%
7275 \renewcommand*\glsxtrsubsequentplfmt}[2]{%
7276   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7277   \ifglsxtrininsertinside \else##2\fi
7278 }%
7279 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
7280   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7281   \ifglsxtrininsertinside \else##2\fi
7282 }%
7283 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
7284   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7285   \ifglsxtrininsertinside \else##2\fi
7286 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7287 \renewcommand*\glsxtrinlinefullformat}[2]{%
7288   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7289   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7290   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7291 }%
7292 \renewcommand*\glsxtrinlinefullplformat}[2]{%
7293   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7294   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7295   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7296 }%
7297 \renewcommand*\Glsxtrinlinefullformat}[2]{%
7298   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7299   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7300   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7301 }%
7302 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
7303   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7304   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7305   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7306 }%

```

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.

```

7307 \renewcommand*\glsxtrfullformat}[2]{%
7308   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7309   \ifglsxtrininsertinside\else##2\fi

```

```

7310 }%
7311 \renewcommand*{\glxtrfullplformat}[2]{%
7312   \glstfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7313   \ifglxtrinsertinside\else##2\fi
7314 }%
7315 \renewcommand*{\Glsxtrfullformat}[2]{%
7316   \glstfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7317   \ifglxtrinsertinside\else##2\fi
7318 }%
7319 \renewcommand*{\Glsxtrfullplformat}[2]{%
7320   \glstfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7321   \ifglxtrinsertinside\else##2\fi
7322 }%
7323 }

```

long-sm Backward compatibility:

```

7324 \@glxtr@deprecated@abbrstyle{long-sm}{long-noshort-sm}

```

noshort-sm-desc The smaller font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

7325 \newabbreviationstyle{long-noshort-sm-desc}%
7326 {%
7327   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7328 }%
7329 {%
7330   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvsmfont{##1}}%
7331   \renewcommand*{\glstfirstabbrvfont}[1]{\glstfirstabbrvsmfont{##1}}%
7332   \renewcommand*{\abbrvpluralsuffix}{\protect\glstxrsmssuffix}%
7333   \renewcommand*{\glstfirstlongfont}[1]{\glstfirstlongdefaultfont{##1}}%
7334   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7335 \renewcommand*{\glstxrsubsequentfmt}[2]{%
7336   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7337   \ifglxtrinsertinside \else##2\fi
7338 }%
7339 \renewcommand*{\glstxrsubsequentplfmt}[2]{%
7340   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7341   \ifglxtrinsertinside \else##2\fi
7342 }%
7343 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7344   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7345   \ifglxtrinsertinside \else##2\fi
7346 }%
7347 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7348   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7349   \ifglxtrinsertinside \else##2\fi
7350 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7351 \renewcommand*{\glxtrinlinefullformat}[2]{%
7352   \glsfirslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7353   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7354   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7355 }%
7356 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7357   \glsfirslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7358   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7359   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7360 }%
7361 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7362   \glsfirslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7363   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7364   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7365 }%
7366 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7367   \glsfirslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7368   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7369   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7370 }%

```

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.

```

7371 \renewcommand*{\glxtrfullformat}[2]{%
7372   \glsfirslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7373   \ifglxtrinsertinside\else##2\fi
7374 }%
7375 \renewcommand*{\glxtrfullplformat}[2]{%
7376   \glsfirslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7377   \ifglxtrinsertinside\else##2\fi
7378 }%
7379 \renewcommand*{\Glsxtrfullformat}[2]{%
7380   \glsfirslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7381   \ifglxtrinsertinside\else##2\fi
7382 }%
7383 \renewcommand*{\Glsxtrfullplformat}[2]{%
7384   \glsfirslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7385   \ifglxtrinsertinside\else##2\fi
7386 }%
7387 }

```

long-desc-sm Backward compatibility:

```

7388 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```

7389 \newabbreviationstyle{short-sm-footnote}%
7390 {%
7391   \renewcommand*{\CustomAbbreviationFields}{%
7392     name={\protect\glsabbrvsmfont{\the\glsshorttok}},

```

```

7393 sort={\the\glsshorttok},
7394 description={\the\glslongtok},%
7395 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
7396 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7397 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
7398 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
7399 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7400 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
7401 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

7402 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7403 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
7404 \glshasattribute{\the\glslabeltok}{regular}%
7405 {%
7406 \glssetattribute{\the\glslabeltok}{regular}{false}%
7407 }%
7408 {}%
7409 }%
7410 }%
7411 {%
7412 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
7413 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
7414 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
7415 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7416 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

7417 \renewcommand*{\glxtrfullformat}[2]{%
7418 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7419 \ifglxtrinsertinside\else##2\fi
7420 \protect\glxtrabbrvfootnote{##1}%
7421 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7422 }%
7423 \renewcommand*{\glxtrfullplformat}[2]{%
7424 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7425 \ifglxtrinsertinside\else##2\fi
7426 \protect\glxtrabbrvfootnote{##1}%
7427 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7428 }%
7429 \renewcommand*{\Glsxtrfullformat}[2]{%
7430 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7431 \ifglxtrinsertinside\else##2\fi
7432 \protect\glxtrabbrvfootnote{##1}%
7433 {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7434 }%
7435 \renewcommand*{\Glsxtrfullplformat}[2]{%
7436 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7437 \ifglxtrinsertinside\else##2\fi

```

```

7438 \protect\glxtrabbrvfootnote{##1}%
7439 {\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
7440 }%

```

The first use full form and the inline full form use the short (long) style.

```

7441 \renewcommand*{\glxtrinlinefullformat}[2]{%
7442 \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7443 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7444 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
7445 }%
7446 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7447 \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7448 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7449 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
7450 }%
7451 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7452 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7453 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7454 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
7455 }%
7456 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7457 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7458 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7459 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
7460 }%
7461 }

```

footnote-sm Backward compatibility:

```

7462 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

sm-postfootnote

```

7463 \newabbreviationstyle{short-sm-postfootnote}%
7464 {%
7465 \renewcommand*{\CustomAbbreviationFields}{%
7466 name={\protect\glsabbrvsmfont{\the\glssshorttok}},
7467 sort={\the\glssshorttok},
7468 description={\the\glslongtok},%
7469 first={\protect\glsfirstabbrvsmfont{\the\glssshorttok}},%
7470 firstplural={\protect\glsfirstabbrvsmfont{\the\glssshortpltok}},%
7471 plural={\protect\glsabbrvsmfont{\the\glssshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

7472 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7473 \csdef{glxtrpostlink\glscategorylabel}{%
7474 \glxtrifwasfirstuse
7475 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

7476      \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
7477      {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%
7478    }%
7479    {}%
7480  }%
7481  \glshasattribute{\the\glslabeltok}{regular}%
7482  {%
7483    \glissetattribute{\the\glslabeltok}{regular}{false}%
7484  }%
7485  {}%
7486 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

7487 \renewcommand*\glxtrsetupfulldefs{%
7488   \let\glxtrifwasfirstuse\@secondoftwo
7489 }%
7490}%
7491{%
7492 \renewcommand*\glxabbrvfont[1]{\glxabbrvsmfont{##1}}%
7493 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7494 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
7495 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
7496 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7497 \renewcommand*\glxtrfullformat[2]{%
7498   \glsfirstabbrvsmfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7499   \ifglxtrininsertinside\else##2\fi
7500 }%
7501 \renewcommand*\glxtrfullplformat[2]{%
7502   \glsfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7503   \ifglxtrininsertinside\else##2\fi
7504 }%
7505 \renewcommand*\Glsxtrfullformat[2]{%
7506   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7507   \ifglxtrininsertinside\else##2\fi
7508 }%
7509 \renewcommand*\Glsxtrfullplformat[2]{%
7510   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7511   \ifglxtrininsertinside\else##2\fi
7512 }%

```

The first use full form and the inline full form use the short (long) style.

```

7513 \renewcommand*\glxtrininlinefullformat[2]{%
7514   \glsfirstabbrvsmfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7515   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7516   \glxtrparen{\glsfirstlongfootnotefont{\glaccesslong{##1}}}%
7517 }%
7518 \renewcommand*\glxtrininlinefullplformat[2]{%
7519   \glsfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

7520 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7521 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
7522 }%
7523 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7524 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7525 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7526 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
7527 }%
7528 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7529 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7530 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7531 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
7532 }%
7533 }

```

postfootnote-sm Backward compatibility:

```

7534 \@glxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glssabbrvemfont`

```

7535 \newcommand*{\glssabbrvemfont}[1]{\emph{#1}}%

```

`\glssfirstabbrvemfont`

```

7536 \newcommand*{\glssfirstabbrvemfont}[1]{\glssabbrvemfont{#1}}%

```

The default short form suffix:

`\glssxtremsuffix`

```

7537 \newcommand*{\glssxtremsuffix}{\glssxtrabbrvpluralsuffix}

```

`\glssfirstlongemfont` Only used by the “long-em” styles.

```

7538 \newcommand*{\glssfirstlongemfont}[1]{\glsslongemfont{#1}}%

```

`\glsslongemfont` Only used by the “long-em” styles.

```

7539 \newcommand*{\glsslongemfont}[1]{\emph{#1}}%

```

`long-short-em` The long form is just set in the default long font.

```

7540 \newabbreviationstyle{long-short-em}%
7541 {%
7542 \renewcommand*{\CustomAbbreviationFields}{%
7543 name={\protect\glssabbrvemfont{\the\glssshorttok}},
7544 sort={\the\glssshorttok},
7545 first={\protect\glssfirstlongdefaultfont{\the\glsslongtok}%
7546 \protect\glssxtrfullsep{\the\glsslabeltok}%
7547 \glssxtrparen{\protect\glssfirstabbrvemfont{\the\glssshorttok}}},%

```

```

7548 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}}%
7549 \protect\glxtrfullsep{\the\glslabeltok}%
7550 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
7551 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}},%
7552 description={\the\glslongtok}}%
7553 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7554 \glsattribute{\the\glslabeltok}{regular}}%
7555 {%
7556 \glssetattribute{\the\glslabeltok}{regular}{false}}%
7557 }%
7558 {}%
7559 }%
7560 }%
7561 {%
7562 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
7563 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
7564 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%

```

Use the default long fonts.

```

7565 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7566 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7567 \renewcommand*\glxtrfullformat[2]{%
7568 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7569 \ifglxtrininsertinside\else##2\fi
7570 \glxtrfullsep{##1}%
7571 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
7572 }%
7573 \renewcommand*\glxtrfullplformat[2]{%
7574 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7575 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7576 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
7577 }%
7578 \renewcommand*\Glsxtrfullformat[2]{%
7579 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7580 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7581 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
7582 }%
7583 \renewcommand*\Glsxtrfullplformat[2]{%
7584 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7585 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7586 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
7587 }%
7588 }

```

g-short-em-desc

```

7589 \newabbreviationstyle{long-short-em-desc}%
7590 {%
7591 \renewcommand*\CustomAbbreviationFields{%

```



```

7592   name={\glxtrlongshortdescname},
7593   sort={\glxtrlongshortdescsort},%
7594   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7595     \protect\glxtrfullsep{\the\glslabeltok}%
7596     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
7597   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7598     \protect\glxtrfullsep{\the\glslabeltok}%
7599     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
7600   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
7601   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
7602 }%

```

Unset the regular attribute if it has been set.

```

7603 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7604   \glshasattribute{\the\glslabeltok}{regular}%
7605   {%
7606     \glissetattribute{\the\glslabeltok}{regular}{false}%
7607   }%
7608 }%
7609 }%
7610 }%
7611 {%

```

As long-short-em style:

```

7612 \GlsXtrUseAbbrStyleFmts{long-short-em}%
7613 }

```

ong-em-short-em

```

7614 \newabbreviationstyle{long-em-short-em}%
7615 {%

```

\glslongemfont is used in the description since \glsdsc doesn't set the style.

```

7616 \renewcommand*{\CustomAbbreviationFields}{%
7617   name={\protect\glsabbrvemfont{\the\glsshorttok}},
7618   sort={\the\glsshorttok},
7619   first={\protect\glsfirstlongemfont{\the\glslongtok}%
7620     \protect\glxtrfullsep{\the\glslabeltok}%
7621     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
7622   firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
7623     \protect\glxtrfullsep{\the\glslabeltok}%
7624     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
7625   plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
7626   description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

7627 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7628   \glshasattribute{\the\glslabeltok}{regular}%
7629   {%
7630     \glissetattribute{\the\glslabeltok}{regular}{false}%
7631   }%

```

```

7632     {}%
7633   }%
7634 }%
7635 {%
7636   \renewcommand*{\abbrvpluralsuffix}{\protect\glstremssuffix}%
7637   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
7638   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
7639   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
7640   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7641   \renewcommand*{\glsxtrfullformat}[2]{%
7642     \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7643     \ifglsxtrinsertinside\else##2\fi
7644     \glsxtrfullsep{##1}%
7645     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
7646   }%
7647   \renewcommand*{\glsxtrfullplformat}[2]{%
7648     \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7649     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7650     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
7651   }%
7652   \renewcommand*{\Glsxtrfullformat}[2]{%
7653     \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7654     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7655     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
7656   }%
7657   \renewcommand*{\Glsxtrfullplformat}[2]{%
7658     \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7659     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7660     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
7661   }%
7662 }

```

m-short-em-desc

```

7663 \newabbreviationstyle{long-em-short-em-desc}%
7664 {%
7665   \renewcommand*{\CustomAbbreviationFields}{%
7666     name={\glsxtrlongshortdescname},
7667     sort={\glsxtrlongshortdescsort},%
7668     first={\protect\glsfirstlongemfont{\the\glslongtok}}%
7669     \protect\glsxtrfullsep{\the\glslabeltok}%
7670     \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
7671     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}}%
7672     \protect\glsxtrfullsep{\the\glslabeltok}%
7673     \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
7674     text={\protect\glsabbrvemfont{\the\glsshorttok}}},%
7675     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
7676   }%

```

Unset the regular attribute if it has been set.

```
7677 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7678   \glshasattribute{\the\glslabeltok}{regular}%
7679   {%
7680     \glissetattribute{\the\glslabeltok}{regular}{false}%
7681   }%
7682   {}%
7683 }%
7684 }%
7685 {%
7686   \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
7687 }
```

short-em-long Now the short (long) version

```
7688 \newabbreviationstyle{short-em-long}%
7689 {%
7690   \renewcommand*{\CustomAbbreviationFields}{%
7691     name={\protect\glsabbrvemfont{\the\glsshorttok}},
7692     sort={\the\glsshorttok},
7693     description={\the\glslongtok},%
7694     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
7695       \protect\glsxtrfullsep{\the\glslabeltok}%
7696       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7697     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
7698       \protect\glsxtrfullsep{\the\glslabeltok}%
7699       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7700     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
7701 }
```

Unset the regular attribute if it has been set.

```
7701 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7702   \glshasattribute{\the\glslabeltok}{regular}%
7703   {%
7704     \glissetattribute{\the\glslabeltok}{regular}{false}%
7705   }%
7706   {}%
7707 }%
7708 }%
7709 {%
```

Mostly as short-long style:

```
7710 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsuffix}%
7711 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
7712 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
7713 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7714 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
7715 \renewcommand*{\glsxtrfullformat}[2]{%
7716   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
7717   \ifglsxtrinsertinside\else##2\fi
```

```

7718 \glxtrfullsep{##1}%
7719 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7720 }%
7721 \renewcommand*{\glxtrfullplformat}[2]{%
7722 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7723 \ifglxtrininsertinside\else##2\fi
7724 \glxtrfullsep{##1}%
7725 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7726 }%
7727 \renewcommand*{\Glsxtrfullformat}[2]{%
7728 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7729 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7730 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7731 }%
7732 \renewcommand*{\Glsxtrfullplformat}[2]{%
7733 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7734 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7735 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7736 }%
7737 }

```

rt-em-long-desc As before but user provides description

```

7738 \newabbreviationstyle{short-em-long-desc}%
7739 {%
7740 \renewcommand*{\CustomAbbreviationFields}{%
7741 name={\glxtrshortlongdescname},
7742 sort={\glxtrshortlongdescsort},
7743 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
7744 \protect\glxtrfullsep{\the\glslabeltok}%
7745 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7746 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
7747 \protect\glxtrfullsep{\the\glslabeltok}%
7748 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7749 text={\protect\glssabbrvemfont{\the\glsshorttok}},%
7750 plural={\protect\glssabbrvemfont{\the\glsshortpltok}}}%
7751 }%

```

Unset the regular attribute if it has been set.

```

7752 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7753 \glshasattribute{\the\glslabeltok}{regular}%
7754 {%
7755 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
7756 }%
7757 }%
7758 }%
7759 }%
7760 {%
7761 \GlsXtrUseAbbrStyleFmts{short-em-long}%
7762 }

```

hort-em-long-em

```
7763 \newabbreviationstyle{short-em-long-em}%
7764 {%
  \glslongemfont is used in the description since \glsdesc doesn't set the style.
7765   \renewcommand*{\CustomAbbreviationFields}{%
7766     name={\protect\glsabbrvemfont{\the\glsshorttok}},
7767     sort={\the\glsshorttok},
7768     description={\protect\glslongemfont{\the\glslongtok}},%
7769     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
7770     \protect\glstrfullsep{\the\glslabeltok}%
7771     \glstrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
7772     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
7773     \protect\glstrfullsep{\the\glslabeltok}%
7774     \glstrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
7775     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
  }
```

Unset the regular attribute if it has been set.

```
7776 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7777   \glsattribute{\the\glslabeltok}{regular}%
7778   {%
7779     \glsattribute{\the\glslabeltok}{regular}{false}%
7780   }%
7781   {}%
7782 }%
7783 }%
7784 {%
7785   \renewcommand*{\abbrvpluralsuffix}{\protect\glstremsuffix}%
7786   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
7787   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
7788   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
7789   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
  }
```

The first use full form and the inline full form are the same for this style.

```
7790 \renewcommand*{\glstrfullformat}[2]{%
7791   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
7792   \ifglstrinsertinside\else##2\fi
7793   \glstrfullsep{##1}%
7794   \glstrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
7795 }%
7796 \renewcommand*{\glstrfullplformat}[2]{%
7797   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
7798   \ifglstrinsertinside\else##2\fi
7799   \glstrfullsep{##1}%
7800   \glstrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
7801 }%
7802 \renewcommand*{\GlsXtrfullformat}[2]{%
7803   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
7804   \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
  }
```

```

7805 \glstrparen{\glfirstlongemfont{\glaccesslong{##1}}}%
7806 }%
7807 \renewcommand*{\Glsxtrfullplformat}[2]{%
7808 \glfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7809 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7810 \glstrparen{\glfirstlongemfont{\glaccesslongpl{##1}}}%
7811 }%
7812 }

```

em-long-em-desc

```

7813 \newabbreviationstyle{short-em-long-em-desc}%
7814 {%
7815 \renewcommand*{\CustomAbbreviationFields}{%
7816 name={\glxtrshortlongdescname},%
7817 sort={\glxtrshortlongdescsort},%
7818 first={\protect\glfirstabbrvemfont{\the\glsshorttok}%
7819 \protect\glxtrfullsep{\the\glslabeltok}%
7820 \glstrparen{\protect\glfirstlongemfont{\the\glslongtok}}},%
7821 firstplural={\protect\glfirstabbrvemfont{\the\glsshortpltok}%
7822 \protect\glxtrfullsep{\the\glslabeltok}%
7823 \glstrparen{\protect\glfirstlongemfont{\the\glslongpltok}}},%
7824 text={\protect\glabbrvemfont{\the\glsshorttok}},%
7825 plural={\protect\glabbrvemfont{\the\glsshortpltok}}%
7826 }%

```

Unset the regular attribute if it has been set.

```

7827 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7828 \glshasattribute{\the\glslabeltok}{regular}%
7829 {%
7830 \glissetattribute{\the\glslabeltok}{regular}{false}%
7831 }%
7832 {}%
7833 }%
7834 }%
7835 {%
7836 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
7837 }

```

short-em

```

7838 \newabbreviationstyle{short-em}%
7839 {%
7840 \renewcommand*{\CustomAbbreviationFields}{%
7841 name={\protect\glabbrvemfont{\the\glsshorttok}},
7842 sort={\the\glsshorttok},
7843 first={\protect\glfirstabbrvemfont{\the\glsshorttok}},
7844 firstplural={\protect\glfirstabbrvemfont{\the\glsshortpltok}},
7845 text={\protect\glabbrvemfont{\the\glsshorttok}},
7846 plural={\protect\glabbrvemfont{\the\glsshortpltok}},
7847 description={\the\glslongtok}}%
7848 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

7849 \glssetattribute{\the\glslabeltok}{regular}{true}}%
7850}%
7851{%
7852 \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
7853 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
7854 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
7855 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7856 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7857 \renewcommand*{\glxstrinlinefullformat}[2]{%
7858 \protect\glsfirstabbrvemfont{\glsaccesssshort{##1}}%
7859 \ifglxstrinsertinside##2\fi}%
7860 \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
7861 \glxstrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7862}%
7863 \renewcommand*{\glxstrinlinefullplformat}[2]{%
7864 \protect\glsfirstabbrvemfont{\glsaccesssshortpl{##1}}%
7865 \ifglxstrinsertinside##2\fi}%
7866 \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
7867 \glxstrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7868}%
7869 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7870 \protect\glsfirstabbrvemfont{\glsaccesssshort{##1}}%
7871 \ifglxstrinsertinside##2\fi}%
7872 \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
7873 \glxstrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7874}%
7875 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7876 \protect\glsfirstabbrvemfont{\glsaccesssshortpl{##1}}%
7877 \ifglxstrinsertinside##2\fi}%
7878 \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
7879 \glxstrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7880}%

```

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.

```

7881 \renewcommand*{\glxstrfullformat}[2]{%
7882 \glsfirstabbrvemfont{\glsaccesssshort{##1}}\ifglxstrinsertinside##2\fi}%
7883 \ifglxstrinsertinside\else##2\fi
7884}%
7885 \renewcommand*{\glxstrfullplformat}[2]{%
7886 \glsfirstabbrvemfont{\glsaccesssshortpl{##1}}\ifglxstrinsertinside##2\fi}%
7887 \ifglxstrinsertinside\else##2\fi
7888}%
7889 \renewcommand*{\Glsxtrfullformat}[2]{%
7890 \glsfirstabbrvemfont{\glsaccesssshort{##1}}\ifglxstrinsertinside##2\fi}%
7891 \ifglxstrinsertinside\else##2\fi
7892}%
7893 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

7894 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7895 \ifglxtrinsertinside\else##2\fi
7896 }%
7897 }

```

short-em-nolong

```

7898 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

7899 \newabbreviationstyle{short-em-desc}%
7900 {%
7901 \renewcommand*{\CustomAbbreviationFields}{%
7902   name={\glxtrshortdescname},
7903   sort={\the\glsshorttok},
7904   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
7905   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
7906   text={\protect\glsabbrvemfont{\the\glsshorttok}},
7907   plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
7908   description={\the\glslongtok}}%
7909 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7910   \glssetAttribute{\the\glslabeltok}{regular}{true}}%
7911 }%
7912 {%
7913 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
7914 \renewcommand*{\glsabbrvfnt}[1]{\glsabbrvemfont{##1}}%
7915 \renewcommand*{\glsfirstabbrvfnt}[1]{\glsfirstabbrvemfont{##1}}%
7916 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7917 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7918 \renewcommand*{\glxtrinlinefullformat}[2]{%
7919   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7920   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7921   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7922 }%
7923 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7924   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7925   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7926   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7927 }%
7928 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7929   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7930   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7931   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7932 }%
7933 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7934   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7935   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7936   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7937 }%

```


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.

```

7938 \renewcommand*{\glxtrfullformat}[2]{%
7939   \glfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7940   \ifglxtrinsertinside\else##2\fi
7941 }%
7942 \renewcommand*{\glxtrfullplformat}[2]{%
7943   \glfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7944   \ifglxtrinsertinside\else##2\fi
7945 }%
7946 \renewcommand*{\Glsxtrfullformat}[2]{%
7947   \glfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7948   \ifglxtrinsertinside\else##2\fi
7949 }%
7950 \renewcommand*{\Glsxtrfullplformat}[2]{%
7951   \glfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7952   \ifglxtrinsertinside\else##2\fi
7953 }%
7954 }

```

-em-nolong-desc

```

7955 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

7956 \newabbreviationstyle{long-noshort-em}%
7957 {%
7958   \renewcommand*{\CustomAbbreviationFields}{%
7959     name={\protect\glsabbrvemfont{\the\glsshorttok}},
7960     sort={\the\glsshorttok},
7961     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7962     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
7963     text={\protect\glslongdefaultfont{\the\glslongtok}},
7964     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7965     description={\the\glslongtok}%
7966   }%
7967   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7968     \glssetAttribute{\the\glslabeltok}{regular}{true}}%
7969 }%
7970 {%
7971   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
7972   \renewcommand*{\glsabbrvf}[1]{\glsabbrvemfont{##1}}%
7973   \renewcommand*{\glsfirstabbrvf}[1]{\glsfirstabbrvemfont{##1}}%
7974   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7975   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7976 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7977   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7978   \ifglxtrinsertinside \else##2\fi

```

```

7979 }%
7980 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7981   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7982   \ifglxtrinsertinside \else##2\fi
7983 }%
7984 \renewcommand*{\Glsxtrsubsequentfml}[2]{%
7985   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7986   \ifglxtrinsertinside \else##2\fi
7987 }%
7988 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7989   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7990   \ifglxtrinsertinside \else##2\fi
7991 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7992 \renewcommand*{\glxtrinlinefullformat}[2]{%
7993   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7994   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7995   \glxtrparen{\protect\glslongfirstabbrvemfont{\glssaccessshort{##1}}}%
7996 }%
7997 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7998   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7999   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8000   \glxtrparen{\protect\glslongfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8001 }%
8002 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8003   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8004   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8005   \glxtrparen{\protect\glslongfirstabbrvemfont{\glssaccessshort{##1}}}%
8006 }%
8007 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8008   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8009   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8010   \glxtrparen{\protect\glslongfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8011 }%

```

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.

```

8012 \renewcommand*{\glxtrfullformat}[2]{%
8013   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8014   \ifglxtrinsertinside\else##2\fi
8015 }%
8016 \renewcommand*{\glxtrfullplformat}[2]{%
8017   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8018   \ifglxtrinsertinside\else##2\fi
8019 }%
8020 \renewcommand*{\Glsxtrfullformat}[2]{%
8021   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8022   \ifglxtrinsertinside\else##2\fi
8023 }%

```

```

8024 \renewcommand*{\Glsxtrfullplformat}[2]{%
8025   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8026   \ifglxtrinsertinside\else##2\fi
8027 }%
8028 }

```

long-em Backward compatibility:

```

8029 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}

```

g-em-noshort-em The short form is explicitly invoked through commands like `\glsshort`.

```

8030 \newabbreviationstyle{long-em-noshort-em}%
8031 {%
8032   \renewcommand*{\CustomAbbreviationFields}{%
8033     name={\protect\glsabbrvfont{\the\glsshorttok}},
8034     sort={\the\glsshorttok},
8035     first={\protect\glsfirstlongemfont{\the\glslongtok}},
8036     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
8037     text={\protect\glslongemfont{\the\glslongtok}},
8038     plural={\protect\glslongemfont{\the\glslongpltok}},%
8039     description={\protect\glslongemfont{\the\glslongtok}}%
8040   }%
8041   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8042     \glssetattribute{\the\glslabeltok}{regular}{true}}%
8043 }%
8044 {%
8045   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8046   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvfont{##1}}%
8047   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvfont{##1}}%
8048   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8049   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8050 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8051   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8052   \ifglxtrinsertinside \else##2\fi
8053 }%
8054 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8055   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8056   \ifglxtrinsertinside \else##2\fi
8057 }%
8058 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8059   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8060   \ifglxtrinsertinside \else##2\fi
8061 }%
8062 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8063   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8064   \ifglxtrinsertinside \else##2\fi
8065 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8066 \renewcommand*{\glxtrinlinefullformat}[2]{%
8067   \glsfirslongemfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8068   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8069   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8070 }%
8071 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8072   \glsfirslongemfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8073   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8074   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8075 }%
8076 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8077   \glsfirslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8078   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8079   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8080 }%
8081 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8082   \glsfirslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8083   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8084   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8085 }%

```

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.

```

8086 \renewcommand*{\glxtrfullformat}[2]{%
8087   \glsfirslongemfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8088   \ifglxtrininsertinside\else##2\fi
8089 }%
8090 \renewcommand*{\glxtrfullplformat}[2]{%
8091   \glsfirslongemfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8092   \ifglxtrininsertinside\else##2\fi
8093 }%
8094 \renewcommand*{\Glsxtrfullformat}[2]{%
8095   \glsfirslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8096   \ifglxtrininsertinside\else##2\fi
8097 }%
8098 \renewcommand*{\Glsxtrfullplformat}[2]{%
8099   \glsfirslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8100   \ifglxtrininsertinside\else##2\fi
8101 }%
8102 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

8103 \newabbreviationstyle{long-em-noshort-em-noreg}%
8104 {%
8105   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

8106 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8107   \glshasattribute{\the\glslabeltok}{regular}%
8108   {%

```

```

8109      \glsetattribute{\the\glslabeltok}{regular}{false}%
8110    }%
8111    {}%
8112  }%
8113}%
8114{%
8115  \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
8116}

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

8117\newabbreviationstyle{long-noshort-em-desc}%
8118{%
8119  \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8120}%
8121{%
8122  \renewcommand*{\abbrvpluralsuffix}{\protect\glstremsuffix}%
8123  \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
8124  \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
8125  \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8126  \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8127  \renewcommand*{\glsxtrsubsequentfmt}[2]{%
8128    \glslongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8129    \ifglsxtrininsertinside \else##2\fi
8130  }%
8131  \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
8132    \glslongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8133    \ifglsxtrininsertinside \else##2\fi
8134  }%
8135  \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8136    \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8137    \ifglsxtrininsertinside \else##2\fi
8138  }%
8139  \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8140    \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8141    \ifglsxtrininsertinside \else##2\fi
8142  }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8143  \renewcommand*{\glsxtrinlinefullformat}[2]{%
8144    \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8145    \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8146    \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshort{##1}}}%
8147  }%
8148  \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8149    \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8150    \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8151    \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshortpl{##1}}}%

```

```

8152 }%
8153 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8154   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8155   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8156   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
8157 }%
8158 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8159   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8160   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8161   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8162 }%

```

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.

```

8163 \renewcommand*{\glxtrfullformat}[2]{%
8164   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8165   \ifglxtrinsertinside\else##2\fi
8166 }%
8167 \renewcommand*{\glxtrfullplformat}[2]{%
8168   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8169   \ifglxtrinsertinside\else##2\fi
8170 }%
8171 \renewcommand*{\Glsxtrfullformat}[2]{%
8172   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8173   \ifglxtrinsertinside\else##2\fi
8174 }%
8175 \renewcommand*{\Glsxtrfullplformat}[2]{%
8176   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8177   \ifglxtrinsertinside\else##2\fi
8178 }%
8179 }

```

long-desc-em Backward compatibility:

```
8180 \@glxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}
```

noshort-em-desc The short form is explicitly invoked through commands like \glsshort. The long form is emphasized.

```

8181 \newabbreviationstyle{long-em-noshort-em-desc}%
8182 {%
8183   \renewcommand*{\CustomAbbreviationFields}{%
8184     name={\protect\protect\glslongemfont{\the\glslongtok}},
8185     sort={\the\glslongtok},
8186     first={\protect\glsfirstlongemfont{\the\glslongtok}},
8187     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
8188     text={\glslongemfont{\the\glslongtok}},
8189     plural={\glslongemfont{\the\glslongpltok}}}%
8190 }%
8191 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8192   \glssetattribute{\the\glslabeltok}{regular}{true}}%

```

```

8193 }%
8194 {%
8195   \renewcommand*{\abbrvpluralsuffix}{\protect\glstremssuffix}%
8196   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
8197   \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvemfont{##1}}%
8198   \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongemfont{##1}}%
8199   \renewcommand*{\glsslongfont}[1]{\glsslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8200   \renewcommand*{\glssxtrsubsequentfmt}[2]{%
8201     \glsslongemfont{\glssaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
8202     \ifglssxtrininsertinside \else##2\fi
8203   }%
8204   \renewcommand*{\glssxtrsubsequentplfmt}[2]{%
8205     \glsslongemfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
8206     \ifglssxtrininsertinside \else##2\fi
8207   }%
8208   \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8209     \glsslongemfont{\Glsaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
8210     \ifglssxtrininsertinside \else##2\fi
8211   }%
8212   \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8213     \glsslongemfont{\Glsaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
8214     \ifglssxtrininsertinside \else##2\fi
8215   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8216   \renewcommand*{\glssxtrininlinefullformat}[2]{%
8217     \glssfirstlongemfont{\glssaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
8218     \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8219     \glssxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshort{##1}}}%
8220   }%
8221   \renewcommand*{\glssxtrininlinefullplformat}[2]{%
8222     \glssfirstlongemfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
8223     \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8224     \glssxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8225   }%
8226   \renewcommand*{\Glsxtrininlinefullformat}[2]{%
8227     \glssfirstlongemfont{\Glsaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
8228     \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8229     \glssxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshort{##1}}}%
8230   }%
8231   \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
8232     \glssfirstlongemfont{\Glsaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
8233     \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8234     \glssxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8235   }%

```

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.

```

8236 \renewcommand*\glstrfullformat}[2]{%
8237   \glstrfirstlongemfont{\glstraccesslong{##1}\ifglstrinsertinside##2\fi}%
8238   \ifglstrinsertinside\else##2\fi
8239 }%
8240 \renewcommand*\glstrfullplformat}[2]{%
8241   \glstrfirstlongemfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
8242   \ifglstrinsertinside\else##2\fi
8243 }%
8244 \renewcommand*\GlsXtrfullformat}[2]{%
8245   \glstrfirstlongemfont{\glstraccesslong{##1}\ifglstrinsertinside##2\fi}%
8246   \ifglstrinsertinside\else##2\fi
8247 }%
8248 \renewcommand*\GlsXtrfullplformat}[2]{%
8249   \glstrfirstlongemfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
8250   \ifglstrinsertinside\else##2\fi
8251 }%
8252 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

8253 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
8254 {%
8255   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%
      Unset the regular attribute if it has been set.
8256   \renewcommand*\GlsXtrPostNewAbbreviation}{%
8257     \glshasattribute{\the\glslabeltok}{regular}%
8258     {%
8259       \glsssetAttribute{\the\glslabeltok}{regular}{false}%
8260     }%
8261   }%
8262 }%
8263 }%
8264 {%
8265   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
8266 }

```

ort-em-footnote

```

8267 \newabbreviationstyle{short-em-footnote}%
8268 {%
8269   \renewcommand*\CustomAbbreviationFields{%
8270     name={\protect\glssabrvemfont{\the\glssshorttok}},
8271     sort={\the\glssshorttok},
8272     description={\the\glslongtok},%
8273     first={\protect\glstrfirstabrvemfont{\the\glssshorttok}%
8274       \protect\glstrabbrvfootnote{\the\glslabeltok}%
8275       {\protect\glstrfirstlongfootnotefont{\the\glslongtok}}},%
8276     firstplural={\protect\glstrfirstabrvemfont{\the\glssshortpltok}%
8277       \protect\glstrabbrvfootnote{\the\glslabeltok}%
8278       {\protect\glstrfirstlongfootnotefont{\the\glslongpltok}}},%
8279     plural={\protect\glssabrvemfont{\the\glssshortpltok}}}%

```


Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

8280 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8281 \glsssetattribute{\the\glslabelltok}{nohyperfirst}{true}%
8282 \glshasattribute{\the\glslabelltok}{regular}%
8283 {%
8284 \glsssetattribute{\the\glslabelltok}{regular}{false}%
8285 }%
8286 {}%
8287 }%
8288 }%
8289 {%
8290 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
8291 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
8292 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvemfont{##1}}%
8293 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongfootnotefont{##1}}%
8294 \renewcommand*{\glsslongfont}[1]{\glsslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

8295 \renewcommand*{\glssxtrfullformat}[2]{%
8296 \glssfirstabbrvemfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
8297 \ifglssxtrininsertinside\else##2\fi
8298 \protect\glssxtrabbrvfootnote{##1}%
8299 {\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
8300 }%
8301 \renewcommand*{\glssxtrfullplformat}[2]{%
8302 \glssfirstabbrvemfont{\glssaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
8303 \ifglssxtrininsertinside\else##2\fi
8304 \protect\glssxtrabbrvfootnote{##1}%
8305 {\glssfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8306 }%
8307 \renewcommand*{\Glsxtrfullformat}[2]{%
8308 \glssfirstabbrvemfont{\Glsaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
8309 \ifglssxtrininsertinside\else##2\fi
8310 \protect\glssxtrabbrvfootnote{##1}%
8311 {\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
8312 }%
8313 \renewcommand*{\Glsxtrfullplformat}[2]{%
8314 \glssfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
8315 \ifglssxtrininsertinside\else##2\fi
8316 \protect\glssxtrabbrvfootnote{##1}%
8317 {\glssfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8318 }%

```

The first use full form and the inline full form use the short (long) style.

```

8319 \renewcommand*{\glssxtrinlinefullformat}[2]{%
8320 \glssfirstabbrvemfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
8321 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8322 \glssxtrparen{\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
8323 }%

```

```

8324 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8325   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8326   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8327   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8328 }%
8329 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8330   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8331   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8332   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8333 }%
8334 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8335   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8336   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8337   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8338 }%
8339 }

```

footnote-em Backward compatibility:

```
8340 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}
```

em-postfootnote

```

8341 \newabbreviationstyle{short-em-postfootnote}%
8342 {%
8343   \renewcommand*{\CustomAbbreviationFields}{%
8344     name={\protect\glssabbrvemfont{\the\glssshorttok}},
8345     sort={\the\glssshorttok},
8346     description={\the\glslongtok},%
8347     first={\protect\glsfirstabbrvemfont{\the\glssshorttok}},%
8348     firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}},%
8349     plural={\protect\glssabbrvemfont{\the\glssshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

8350 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8351   \csdef{glxtrpostlink\glscategorylabel}{%
8352     \glxtrifwasfirstuse
8353     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8354       \glxtrdopostpunc{\protect\glxtrabbrvf footnote{\glslabel}%
8355       {\glsfirstlongfootnotefont{\glssentrylong{\glslabel}}}%
8356     }%
8357   }%
8358 }%
8359 \glshasattribute{\the\glslabeltok}{regular}%
8360 {%
8361   \glsssetattribute{\the\glslabeltok}{regular}{false}%
8362 }%
8363 }%

```

8364 }%

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
8365 \renewcommand*\glxtrsetupfulldefs}{%
8366   \let\glxtrifwasfirstuse\@secondoftwo
8367 }%
8368 }%
8369 {%
8370 \renewcommand*\abbrvpluralsuffix{\protect\glxtremsuffix}%
8371 \renewcommand*\glabbrvfont[1]{\glabbrvemfont{##1}}%
8372 \renewcommand*\glfirstabbrvfont[1]{\glfirstabbrvemfont{##1}}%
8373 \renewcommand*\glfirstlongfont[1]{\glfirstlongfootnotefont{##1}}%
8374 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
8375 \renewcommand*\glxtrfullformat}[2]{%
8376   \glfirstabbrvemfont{\glaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
8377   \ifglxtrininsertinside\else##2\fi
8378 }%
8379 \renewcommand*\glxtrfullplformat}[2]{%
8380   \glfirstabbrvemfont{\glaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
8381   \ifglxtrininsertinside\else##2\fi
8382 }%
8383 \renewcommand*\Glsxtrfullformat}[2]{%
8384   \glfirstabbrvemfont{\Glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
8385   \ifglxtrininsertinside\else##2\fi
8386 }%
8387 \renewcommand*\Glsxtrfullplformat}[2]{%
8388   \glfirstabbrvemfont{\Glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
8389   \ifglxtrininsertinside\else##2\fi
8390 }%
```

The first use full form and the inline full form use the short (long) style.

```
8391 \renewcommand*\glxtrininlinefullformat}[2]{%
8392   \glfirstabbrvemfont{\glaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
8393   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8394   \glxtrparen{\glfirstlongfootnotefont{\glaccesslong{##1}}}%
8395 }%
8396 \renewcommand*\glxtrininlinefullplformat}[2]{%
8397   \glfirstabbrvemfont{\glaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
8398   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8399   \glxtrparen{\glfirstlongfootnotefont{\glaccesslongpl{##1}}}%
8400 }%
8401 \renewcommand*\Glsxtrininlinefullformat}[2]{%
8402   \glfirstabbrvemfont{\Glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
8403   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8404   \glxtrparen{\glfirstlongfootnotefont{\glaccesslong{##1}}}%
8405 }%
8406 \renewcommand*\Glsxtrininlinefullplformat}[2]{%
8407   \glfirstabbrvemfont{\Glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
8408   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8409   \glxtrparen{\glfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8410 }%
```

```

8408      \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8409      \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8410    }%
8411 }

```

postfootnote-em Backward compatibility:

```

8412 \@glxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}

```

1.6.6 Predefined Styles (User Parentheses Hook)

These styles allow the user to adjust the parenthetical forms. These styles all test for the existence of the field given by:

glxtruserfield Default is the useri field.

```

8413 \newcommand*{\glxtruserfield}{useri}

```

glxtruserparen The format of the parenthetical information. The first argument is the long/short form. The second argument is the entry's label. If \glscurrentfieldvalue has been defined, then we have at least glossaries v4.23, which makes it easier for the user to adjust this.

```

8414 \ifdef\glscurrentfieldvalue
8415 {
8416   \newcommand*{\glxtruserparen}[2]{%
8417     \glxtrfullsep{#2}%
8418     \glxtrparen
8419       {#1\ifglshasfield{\glxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
8420   }
8421 }
8422 {
8423   \newcommand*{\glxtruserparen}[2]{%
8424     \glxtrfullsep{#2}%
8425     \glxtrparen
8426       {#1\ifglshasfield{\glxtruserfield}{#2}{, \@glo@thisvalue}{}}%
8427   }
8428 }

```

Font used for short form:

lsabbrvuserfont

```

8429 \newcommand*{\glsabbrvuserfont}[1]{\glsabbrvdefaultfont{#1}}

```

Font used for short form on first use:

stabbrvuserfont

```

8430 \newcommand*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}

```

Font used for long form:

glslonguserfont

```

8431 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}

```

Font used for long form on first use:

rstlonguserfont

```
8432 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
8433 \newcommand*{\glsxtrusersuffix}{\glsxtrabbrvpluralsuffix}
```

long-short-user

```
8434 \newabbreviationstyle{long-short-user}%
8435 {%
8436   \renewcommand*{\CustomAbbreviationFields}{%
8437     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
8438     sort={\the\glsshorttok},
8439     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
8440       \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
8441       {\the\glslabeltok}},%
8442     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
8443       \protect\glsxtruserparen
8444       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
8445     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
8446     description={\protect\glslonguserfont{\the\glslongtok}}}%
8447   }
```

Unset the regular attribute if it has been set.

```
8447   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8448     \glshasattribute{\the\glslabeltok}{regular}%
8449     {%
8450       \glissetattribute{\the\glslabeltok}{regular}{false}%
8451     }%
8452   }%
8453 }%
8454 }%
8455 }
```

In case the user wants to mix and match font styles, these are redefined here.

```
8456 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
8457 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
8458 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
8459 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
8460 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8461 \renewcommand*{\glsxtrfullformat}[2]{%
8462   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8463   \ifglsxtrinsertinside\else##2\fi
8464   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
8465 }%
8466 \renewcommand*{\glsxtrfullplformat}[2]{%
8467   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8468   \ifglsxtrinsertinside\else##2\fi
8469   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
8470 }
```

```

8467 \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8468 \ifglxtrinsertinside\else##2\fi
8469 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}{##1}%
8470 }%
8471 \renewcommand*{\Glsxtrfullformat}[2]{%
8472 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8473 \ifglxtrinsertinside\else##2\fi
8474 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}{##1}%
8475 }%
8476 \renewcommand*{\Glsxtrfullplformat}[2]{%
8477 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8478 \ifglxtrinsertinside\else##2\fi
8479 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}{##1}%
8480 }%
8481 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

8482 \newabbreviationstyle{long-postshort-user}%
8483 {%
8484 \renewcommand*{\CustomAbbreviationFields}{%
8485 name={\protect\glsabbrvuserfont{\the\glsshorttok}},
8486 sort={\the\glsshorttok},
8487 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
8488 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
8489 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
8490 description={\protect\glslonguserfont{\the\glslongtok}}}%
8491 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8492 \csdef{glxtrpostlink\glscategorylabel}{%
8493 \glxtrifwasfirstuse
8494 {%
8495 \glxtruserparen
8496 {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
8497 {\glslabel}%
8498 }%
8499 }%
8500 }%
8501 \glshasattribute{\the\glslabeltok}{regular}%
8502 {%
8503 \glssetattribute{\the\glslabeltok}{regular}{false}%
8504 }%
8505 }%
8506 }%
8507 }%
8508 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8509 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
8510 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
8511 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%

```

```

8512 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
8513 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

8514 \renewcommand*{\glsxtrfullformat}[2]{%
8515   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8516   \ifglsxtrinsertinside\else##2\fi
8517 }%
8518 \renewcommand*{\glsxtrfullplformat}[2]{%
8519   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8520   \ifglsxtrinsertinside\else##2\fi
8521 }%
8522 \renewcommand*{\Glsxtrfullformat}[2]{%
8523   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8524   \ifglsxtrinsertinside\else##2\fi
8525 }%
8526 \renewcommand*{\Glsxtrfullplformat}[2]{%
8527   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8528   \ifglsxtrinsertinside\else##2\fi
8529 }%

```

In-line format:

```

8530 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8531   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8532   \ifglsxtrinsertinside\else##2\fi
8533   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
8534 }%
8535 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8536   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8537   \ifglsxtrinsertinside\else##2\fi
8538   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
8539 }%
8540 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8541   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8542   \ifglsxtrinsertinside\else##2\fi
8543   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
8544 }%
8545 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8546   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8547   \ifglsxtrinsertinside\else##2\fi
8548   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
8549 }%
8550 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

8551 \newabbreviationstyle{long-postshort-user-desc}%
8552 {%
8553   \renewcommand*{\CustomAbbreviationFields}{%
8554     name={\protect\glslonguserfont{\the\glslongtok}}%
8555     \protect\glsxtruserparen

```

```

8556         {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}},
8557     sort={\the\glslongtok},
8558     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
8559     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

8560     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
8561     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
8562 }%
8563 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8564     \csdef{glsxtrpostlink\glscategorylabel}{%
8565         \glsxtrifwasfirstuse
8566         {%
8567             \glsxtruserparen
8568             {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
8569             {\glslabel}}%
8570         }%
8571     }%
8572 }%
8573 \glsattribute{\the\glslabeltok}{regular}%
8574 {%
8575     \glssetattribute{\the\glslabeltok}{regular}{false}%
8576 }%
8577 {}%
8578 }%
8579 }%
8580 {%
8581     \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
8582 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

8583 \newabbreviationstyle{short-postlong-user}%
8584 {%
8585     \renewcommand*{\CustomAbbreviationFields}{%
8586         name={\protect\glsabbrvuserfont{\the\glsshorttok}},
8587         sort={\the\glsshorttok},
8588         first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
8589         firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

8590         plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
8591         description={\protect\glsfirstlonguserfont{\the\glslongtok}}}%
8592 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8593     \csdef{glsxtrpostlink\glscategorylabel}{%
8594         \glsxtrifwasfirstuse
8595         {%
8596             \glsxtruserparen
8597             {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
8598             {\glslabel}}%
8599         }%
8600     }%
8601 }%

```



```

8602 \glshasattribute{\the\glslabeltok}{regular}%
8603 {%
8604 \glissetattribute{\the\glslabeltok}{regular}{false}%
8605 }%
8606 {}%
8607 }%
8608 }%
8609 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8610 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
8611 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
8612 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
8613 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
8614 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

8615 \renewcommand*{\glsxtrfullformat}[2]{%
8616 \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8617 \ifglsxtrininsertinside\else##2\fi
8618 }%
8619 \renewcommand*{\glsxtrfullplformat}[2]{%
8620 \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8621 \ifglsxtrininsertinside\else##2\fi
8622 }%
8623 \renewcommand*{\Glsxtrfullformat}[2]{%
8624 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8625 \ifglsxtrininsertinside\else##2\fi
8626 }%
8627 \renewcommand*{\Glsxtrfullplformat}[2]{%
8628 \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8629 \ifglsxtrininsertinside\else##2\fi
8630 }%

```

In-line format:

```

8631 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8632 \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8633 \ifglsxtrininsertinside\else##2\fi
8634 \glsxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
8635 }%
8636 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8637 \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8638 \ifglsxtrininsertinside\else##2\fi
8639 \glsxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
8640 }%
8641 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8642 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8643 \ifglsxtrininsertinside\else##2\fi
8644 \glsxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
8645 }%

```

```

8646 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8647   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8648   \ifglxtrinsertinside\else##2\fi
8649   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
8650 }%
8651 }

```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```

8652 \newabbreviationstyle{short-postlong-user-desc}%
8653 {%
8654   \renewcommand*{\CustomAbbreviationFields}{%
8655     name={\protect\glsabbrvuserfont{\the\glsshorttok}%
8656       \protect\glxtruserparen
8657         {\protect\glslonguserfont{\the\glslongpltok}}%
8658         {\the\glslabeltok}},
8659     sort={\the\glsshorttok},
8660     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
8661     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
8662     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
8663     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
8664   }%
8665   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8666     \csdef{glxtrpostlink\glscategorylabel}{%
8667       \glxtrifwasfirstuse
8668       {%
8669         \glxtruserparen
8670         {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
8671         {\glslabel}}%
8672       }%
8673     }%
8674   }%
8675   \glshasattribute{\the\glslabeltok}{regular}%
8676   {%
8677     \glissetattribute{\the\glslabeltok}{regular}{false}%
8678   }%
8679   {}%
8680 }%
8681 }%
8682 {%
8683   \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
8684 }

```

short-user-desc

```

8685 \newabbreviationstyle{long-short-user-desc}%
8686 {%
8687   \renewcommand*{\CustomAbbreviationFields}{%
8688     name={\glxtrlongshortdescname},
8689     sort={\glxtrlongshortdescsort},%

```

```

8690 first={\protect\glsfirstlonguserfont{\the\glslongtok}%
8691 \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
8692 {\the\glslabeltok}},%
8693 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
8694 \protect\glsxtruserparen
8695 {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
8696 text={\protect\glsabbrvfont{\the\glsshorttok}},%
8697 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
8698 }%

```

Unset the regular attribute if it has been set.

```

8699 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8700 \glshasattribute{\the\glslabeltok}{regular}%
8701 {%
8702 \glissetattribute{\the\glslabeltok}{regular}{false}%
8703 }%
8704 {}%
8705 }%
8706 }%
8707 {%
8708 \GlsXtrUseAbbrStyleFmts{long-short-user}%
8709 }

```

short-long-user

```

8710 \newabbreviationstyle{short-long-user}%
8711 {%
8712 \glslonguserfont is used in the description since \glsdesc doesn't set the style.
8712 \renewcommand*{\CustomAbbreviationFields}{%
8713 name={\protect\glsabbrvuserfont{\the\glsshorttok}},
8714 sort={\the\glsshorttok},
8715 description={\protect\glslonguserfont{\the\glslongtok}},%
8716 first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
8717 \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
8718 {\the\glslabeltok}},%
8719 firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
8720 \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
8721 {\the\glslabeltok}},%
8722 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8723 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8724 \glshasattribute{\the\glslabeltok}{regular}%
8725 {%
8726 \glissetattribute{\the\glslabeltok}{regular}{false}%
8727 }%
8728 {}%
8729 }%
8730 }%
8731 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
8732 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
8733 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
8734 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
8735 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
8736 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8737 \renewcommand*{\glxtrfullformat}[2]{%
8738   \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8739   \ifglxtrininsertinside\else##2\fi
8740   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
8741 }%
8742 \renewcommand*{\glxtrfullplformat}[2]{%
8743   \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8744   \ifglxtrininsertinside\else##2\fi
8745   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
8746 }%
8747 \renewcommand*{\Glsxtrfullformat}[2]{%
8748   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8749   \ifglxtrininsertinside\else##2\fi
8750   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
8751 }%
8752 \renewcommand*{\Glsxtrfullplformat}[2]{%
8753   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8754   \ifglxtrininsertinside\else##2\fi
8755   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
8756 }%
8757 }
```

-long-user-desc

```
8758 \newabbreviationstyle{short-long-user-desc}%
8759 {%
8760   \renewcommand*{\CustomAbbreviationFields}{%
8761     name={\glxtrshortlongdescname},
8762     sort={\glxtrshortlongdescsort},%
8763     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
8764     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
8765     {\the\glslabeltok}},%
8766     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
8767     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%
8768     {\the\glslabeltok}},%
8769     text={\protect\glsabbrvfont{\the\glsshorttok}},%
8770     plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
8771   }%
```

Unset the regular attribute if it has been set.

```
8772 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8773   \glshasattribute{\the\glslabeltok}{regular}%
```

```

8774    {%
8775        \glsetattribute{\the\glslabeltok}{regular}{false}%
8776    }%
8777    {}%
8778 }%
8779}%
8780{%
8781    \GlsXtrUseAbbrStyleFmts{short-long-user}%
8782 }

```

1.6.7 Predefined Styles (Hyphen)

These styles are designed to work with the `keywords` attribute. They check if the inserted material (provided by the final optional argument of commands like `\gls`) starts with a hyphen. If it does, the insert is added to the parenthetical material. Note that commands like `\glsxtrlongset` `\glsinsert` to empty with the entire link-text stored in `\glscustomtext`.

`\glsxtrifhyphenstart` Checks if the argument starts with a hyphen. The argument may be `\glsinsert` so check for that and expand.

```

8783 \newrobustcmd*{\glsxtrifhyphenstart}[3]{%
8784     \ifx\glsinsert#1\relax
8785     \expandafter\@glsxtrifhyphenstart#1\relax\relax
8786     \@end@glsxtrifhyphenstart{#2}{#3}%
8787 }else
8788     \@glsxtrifhyphenstart#1\relax\relax\@end@glsxtrifhyphenstart{#2}{#3}%
8789 \fi
8790 }

```

`\glsxtrifhyphenstart`

```

8791 \def\@glsxtrifhyphenstart#1#2\@end@glsxtrifhyphenstart#3#4{%
8792     \ifx-#1\relax#3\else #4\fi
8793 }

```

`\glsxtrlonghyphenshort`

```
\glsxtrlonghyphenshort{<label>}{<long>}{<short>}{<insert>}
```

The `<long>` and `<short>` arguments may be the plural form. The `<long>` argument may also be the first letter uppercase form.

```
8794 \newcommand*{\glsxtrlonghyphenshort}[4]{%
```

Grouping is needed to localise the redefinitions.

```
8795 {%
```

If `<insert>` starts with a hyphen, redefine `\glsxtrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.) No change is made to `\glsxtrwordsep` if `<insert>` doesn't start with a hyphen.

```
8796     \glsxtrifhyphenstart{#4}{\def\glsxtrwordsep{-}}{}%
```

```

8797 \glsfirstlonghyphenfont{#2\ifglxtrinsertinside{#4}\fi}%
8798 \ifglxtrinsertinside\else{#4}\fi
8799 \glxtrfullsep{#1}%
8800 \glxtrparen{\glsfirstabbrvhyphenfont{#3\ifglxtrinsertinside{#4}\fi}%
8801 \ifglxtrinsertinside\else{#4}\fi}%
8802 }%
8803 }

```

abbrvhyphenfont

```
8804 \newcommand*{\glsabbrvhyphenfont}{\glsabbrvdefaultfont}%
```

abbrvhyphenfont

```
8805 \newcommand*{\glsfirstabbrvhyphenfont}{\glsabbrvhyphenfont}%
```

slonghyphenfont

```
8806 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%
```

tlonghyphenfont

```
8807 \newcommand*{\glsfirstlonghyphenfont}{\glslonghyphenfont}%
```

The default short form suffix:

xtrhyphensuffix

```
8808 \newcommand*{\glxtrhyphensuffix}{\glxtrabbrvpluralsuffix}
```

en-short-hyphen Designed for use with the markwords attribute.

```

8809 \newabbreviationstyle{long-hyphen-short-hyphen}%
8810 {%
8811 \renewcommand*{\CustomAbbreviationFields}{%
8812 name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
8813 sort={\the\glsshorttok},
8814 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
8815 \protect\glxtrfullsep{\the\glslabeltok}%
8816 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
8817 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
8818 \protect\glxtrfullsep{\the\glslabeltok}%
8819 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
8820 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
8821 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

8822 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8823 \glshasattribute{\the\glslabeltok}{regular}%
8824 {%
8825 \glssetattribute{\the\glslabeltok}{regular}{false}%
8826 }%
8827 {}%
8828 }%
8829 }%

```

```

8830 {%
8831   \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
8832   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
8833   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
8834   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
8835   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8836   \renewcommand*{\glxtrfullformat}[2]{%
8837     \glxtrlonghyphenshort{##1}{\glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
8838   }%
8839   \renewcommand*{\glxtrfullplformat}[2]{%
8840     \glxtrlonghyphenshort{##1}{\glsaccesslongpl{##1}}%
8841     {\glsaccessshortpl{##1}}{##2}%
8842   }%
8843   \renewcommand*{\Glsxtrfullformat}[2]{%
8844     \glxtrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
8845   }%
8846   \renewcommand*{\Glsxtrfullplformat}[2]{%
8847     \glxtrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
8848     {\glsaccessshortpl{##1}}{##2}%
8849   }%
8850 }

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

8851 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
8852 {%
8853   \renewcommand*{\CustomAbbreviationFields}{%
8854     name={\glxtrlongshortdescname},
8855     sort={\glxtrlongshortdescsort},
8856     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%
8857     \protect\glxtrfullsep{\the\glslabeltok}%
8858     \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
8859     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%
8860     \protect\glxtrfullsep{\the\glslabeltok}%
8861     \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
8862     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
8863     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
8864   }%

```

Unset the regular attribute if it has been set.

```

8865   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8866     \glshasattribute{\the\glslabeltok}{regular}%
8867     {%
8868       \glissetattribute{\the\glslabeltok}{regular}{false}%
8869     }%
8870   }%
8871 }%
8872 }%
8873 {%

```

```

8874 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
8875 }

```

onghyphennoshort

```
\glstrlonghyphennoshort{<label>}{<long>}{<insert>}
```

```

8876 \newcommand*{\glstrlonghyphennoshort}[3]{%

```

Grouping is needed to localise the redefinitions.

```
8877 {%
```

If *<insert>* starts with a hyphen, redefine `\glstrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.) No change is made to `\glstrwordsep` if *<insert>* doesn't start with a hyphen.

```

8878 \glstrifhyphenstart{#3}{\def\glstrwordsep{-}}{%
8879 \glstrfirstlonghyphenfont{#2\ifglstrinsertinside{#3}\fi}%
8880 \ifglstrinsertinside\else{#3}\fi
8881 }%
8882 }

```

short-desc-noreg

This version doesn't show the short form (except explicitly with `\glstrshort`). Since `\glstrshort` doesn't support the hyphen switch, the short form just uses the default short-form font command. This style won't work with the regular as the regular form isn't flexible enough.

```

8883 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
8884 {%
8885 \renewcommand*{\CustomAbbreviationFields}{%
8886 name={\protect\protect\glslonghyphenfont{\the\glslongtok}},
8887 sort={\expandonce\glstrorglong},
8888 first={\protect\glstrfirstlonghyphenfont{\the\glslongtok}},%
8889 firstplural={\protect\glstrfirstlonghyphenfont{\the\glslongpltok}},%
8890 plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
8891 }%

```

Unset the regular attribute if it has been set.

```

8892 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8893 \glshasattribute{\the\glslabeltok}{regular}%
8894 {%
8895 \glissetattribute{\the\glslabeltok}{regular}{false}%
8896 }%
8897 {}%
8898 }%
8899 }%
8900 {%
8901 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%

```

In case the user wants to mix and match font styles, these are redefined here.

```
8902 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
```



```

8903 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
8904 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
8905 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonghyphenfont{##1}}%
8906 \renewcommand*\glslongfont[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8907 \renewcommand*\glsxtrsubsequentfmt[2]{%
8908   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
8909 }%
8910 \renewcommand*\glsxtrsubsequentplfmt[2]{%
8911   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
8912 }%
8913 \renewcommand*\Glsxtrsubsequentfmt[2]{%
8914   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
8915 }%
8916 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
8917   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
8918 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8919 \renewcommand*\glsxtrinlinefullformat[2]{%
8920   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
8921   \glsxtrfullsep{##1}%
8922   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8923 }%
8924 \renewcommand*\glsxtrinlinefullplformat[2]{%
8925   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
8926   \glsxtrfullsep{##1}%
8927   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8928 }%
8929 \renewcommand*\Glsxtrinlinefullformat[2]{%
8930   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
8931   \glsxtrfullsep{##1}%
8932   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8933 }%
8934 \renewcommand*\Glsxtrinlinefullplformat[2]{%
8935   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
8936   \glsxtrfullsep{##1}%
8937   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8938 }%

```

The first use full form only displays the long form.

```

8939 \renewcommand*\glsxtrfullformat[2]{%
8940   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
8941 }%
8942 \renewcommand*\glsxtrfullplformat[2]{%
8943   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
8944 }%
8945 \renewcommand*\Glsxtrfullformat[2]{%
8946   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
8947 }%

```

```

8948 \renewcommand*{\Glsxtrfullplformat}[2]{%
8949   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
8950 }%
8951 }

```

n-noshort-noreg It doesn't really make a great deal of sense to have a long-only style that doesn't have a description (unless no glossary is required), but the best course of action here is to use the short form as the name and the long form as the description.

```

8952 \newabbreviationstyle{long-hyphen-noshort-noreg}%
8953 {%
8954   \renewcommand*{\CustomAbbreviationFields}{%
8955     name={\protect\glsabbrvfont{\the\glsshorttok}},
8956     sort={\the\glsshorttok},
8957     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
8958     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
8959     text={\protect\glslonghyphenfont{\the\glslongtok}},%
8960     plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
8961     description={\the\glslongtok}%
8962   }%

```

Unset the regular attribute if it has been set.

```

8963 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8964   \glsattribute{\the\glslabeltok}{regular}%
8965   {%
8966     \glssetattribute{\the\glslabeltok}{regular}{false}%
8967   }%
8968   {}%
8969 }%
8970 }%
8971 {%
8972   \GlsXtrUseAbbrStyleFmts{long-desc}%
8973 }

```

glsxtrlonghyphen `\glsxtrlonghyphen{<long>}{<label>}{<insert>}`

Used by long-hyphen-postshort-hyphen. The *<insert>* is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```

8974 \newcommand*{\glsxtrlonghyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

8975   {%
8976     \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
8977     \glsfirstlonghyphenfont{#1}%
8978   }%
8979 }

```

posthyphenshort

`\glxtrposthyphenshort{<label>}{<insert>}`

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like `\glxtrlonghyphenshort` but omits the *<long>* part. This always uses the singular short form.

```
8980 \newcommand*{\glxtrposthyphenshort}[2]{%
8981   {%
8982     \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}}%
8983     \ifglxtrininsertinside{\glsfirstlonghyphenfont{#2}}\else{#2}\fi
8984     \glxtrfullsep{#1}%
8985     \glxtrparen
8986     {\glsfirstabbrvhyphenfont{\glsenentryshort{#1}}\ifglxtrininsertinside{#2}\fi}%
8987     \ifglxtrininsertinside\else{#2}\fi
8988   }%
8989 }%
8990 }
```

hyphensubsequent

`\glxtrposthyphensubsequent{<label>}{<insert>}`

Format in the post-link hook for subsequent use. The label is ignored by default.

```
8991 \newcommand*{\glxtrposthyphensubsequent}[2]{%
8992   \glssabbrvfont{\ifglxtrininsertinside {#2}\fi}%
8993   \ifglxtrininsertinside \else{#2}\fi
8994 }
```

postshort-hyphen

Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```
8995 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
8996 {%
8997   \renewcommand*{\CustomAbbreviationFields}{%
8998     name={\protect\glssabbrvhyphenfont{\the\glssshorttok}},
8999     sort={\the\glssshorttok},
9000     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9001     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9002     plural={\protect\glssabbrvhyphenfont{\the\glssshortpltok}},%
9003     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
9004   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9005     \csdef{glxtrpostlink\glscategorylabel}{%
9006       \glxtrifwasfirstuse
9007       {%
9008         \glxtrposthyphenshort{\glslabel}{\glssinsert}%
9009       }%
9010     }%
```

Put the insertion into the post-link:

```

9011      \glxtrposthyphensubsequent{\glslabel}{\glinsert}%
9012    }%
9013  }%
9014    \glshasattribute{\the\glslabeltok}{regular}%
9015    {%
9016      \glissetattribute{\the\glslabeltok}{regular}{false}%
9017    }%
9018    {}%
9019  }%
9020}%
9021{%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9022 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
9023 \renewcommand*\glsabbrvfont[1]{\glsabbrvhyphenfont{##1}}%
9024 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvhyphenfont{##1}}%
9025 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonghyphenfont{##1}}%
9026 \renewcommand*\glslongfont[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

9027 \renewcommand*\glxtrsubsequentfmt[2]{%
9028   \glsabbrvfont{\glsaccessshort{##1}}%
9029 }%
9030 \renewcommand*\glxtrsubsequentplfmt[2]{%
9031   \glsabbrvfont{\glsaccessshortpl{##1}}%
9032 }%
9033 \renewcommand*\Glsxtrsubsequentfmt[2]{%
9034   \glsabbrvfont{\Glsaccessshort{##1}}%
9035 }%
9036 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
9037   \glsabbrvfont{\Glsaccessshortpl{##1}}%
9038 }%

```

First use full form:

```

9039 \renewcommand*\glxtrfullformat[2]{%
9040   \glxtrlonghyphen{\glsaccesslong{##1}}{##1}{##2}%
9041 }%
9042 \renewcommand*\glxtrfullplformat[2]{%
9043   \glxtrlonghyphen{\glsaccesslongpl{##1}}{##1}{##2}%
9044 }%
9045 \renewcommand*\Glsxtrfullformat[2]{%
9046   \glxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
9047 }%
9048 \renewcommand*\Glsxtrfullplformat[2]{%
9049   \glxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
9050 }%

```

In-line format.

```

9051 \renewcommand*\glxtrinlinefullformat[2]{%
9052   \glsfirstlonghyphenfont{\glsaccesslong{##1}}%
9053   \ifglxtrininsertinside{##2}\fi}%

```

```

9054 \ifglxtrinsertinside \else{##2}\fi
9055 }%
9056 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9057 \glsfirslonghyphenfont{\glssaccesslongpl{##1}%
9058 \ifglxtrinsertinside{##2}\fi}%
9059 \ifglxtrinsertinside \else{##2}\fi
9060 }%
9061 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9062 \glsfirslonghyphenfont{\Glsaccesslong{##1}%
9063 \ifglxtrinsertinside{##2}\fi}%
9064 \ifglxtrinsertinside \else{##2}\fi
9065 }%
9066 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9067 \glsfirslonghyphenfont{\Glsaccesslongpl{##1}%
9068 \ifglxtrinsertinside{##2}\fi}%
9069 \ifglxtrinsertinside \else{##2}\fi
9070 }%
9071 }

```

ort-hyphen-desc Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

9072 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
9073 {%
9074 \renewcommand*{\CustomAbbreviationFields}{%
9075 name={\glxtrlongshortdescname},
9076 sort={\glxtrlongshortdescsort},%
9077 first={\protect\glsfirslonghyphenfont{\the\glslongtok}},%
9078 firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}},%
9079 text={\protect\glssabbrvhyphenfont{\the\glssshorttok}},%
9080 plural={\protect\glssabbrvhyphenfont{\the\glssshortpltok}}}%
9081 }%
9082 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9083 \csdef{glxtrpostlink\glscategorylabel}{%
9084 \glxtrifwasfirstuse
9085 {%
9086 \glxtrposthyphenshort{\glslabel}{\glssinsert}%
9087 }%
9088 }%

```

Put the insertion into the post-link:

```

9089 \glxtrposthyphensubsequent{\glslabel}{\glssinsert}%
9090 }%
9091 }%
9092 \glshasattribute{\the\glslabeltok}{regular}%
9093 {%
9094 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9095 }%
9096 {}%
9097 }%
9098 }%
9099 {%

```

```

9100 \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
9101 }

```

rshorthyphenlong

```
\glxtrshorthyphenlong{<label>}{<short>}{<long>}{<insert>}
```

The *<long>* and *<short>* arguments may be the plural form. The *<long>* argument may also be the first letter uppercase form.

```
9102 \newcommand*{\glxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
9103 {%
```

If *<insert>* starts with a hyphen, redefine `\glxtrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.)

```

9104 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{}%
9105 \glsfirstabbrvhyphenfont{#2\ifglxtrininsertinside{#4}\fi}%
9106 \ifglxtrininsertinside\else{#4}\fi
9107 \glxtrfullsep{#1}%
9108 \glxtrparen{\glsfirstlonghyphenfont{#3\ifglxtrininsertinside{#4}\fi}%
9109 \ifglxtrininsertinside\else{#4}\fi}%
9110 }%
9111 }

```

hen-long-hyphen

Designed for use with the `markwords` attribute.

```

9112 \newabbreviationstyle{short-hyphen-long-hyphen}%
9113 {%
9114 \renewcommand*{\CustomAbbreviationFields}{%
9115 name={\protect\glssabbrvhyphenfont{\the\glssshorttok}},
9116 sort={\the\glssshorttok},
9117 first={\protect\glsfirstabbrvhyphenfont{\the\glssshorttok}}%
9118 \protect\glxtrfullsep{\the\glslabeltok}%
9119 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
9120 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glssshortpltok}}%
9121 \protect\glxtrfullsep{\the\glslabeltok}%
9122 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
9123 plural={\protect\glssabbrvhyphenfont{\the\glssshortpltok}}},%
9124 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

9125 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9126 \glshasattribute{\the\glslabeltok}{regular}%
9127 {%
9128 \glissetattribute{\the\glslabeltok}{regular}{false}%
9129 }%
9130 {}%
9131 }%

```

```

9132 }%
9133 {%
9134   \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
9135   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9136   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9137   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9138   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9139   \renewcommand*{\glxtrfullformat}[2]{%
9140     \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
9141   }%
9142   \renewcommand*{\glxtrfullplformat}[2]{%
9143     \glxtrshorthyphenlong{##1}%
9144     {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
9145   }%
9146   \renewcommand*{\Glsxtrfullformat}[2]{%
9147     \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
9148   }%
9149   \renewcommand*{\Glsxtrfullplformat}[2]{%
9150     \glxtrshorthyphenlong{##1}%
9151     {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
9152   }%
9153 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

9154 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
9155 {%
9156   \renewcommand*{\CustomAbbreviationFields}{%
9157     name={\glxtrshortlongdescname},
9158     sort={\glxtrshortlongdescsort},
9159     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%
9160     \protect\glxtrfullsep{\the\glslabeltok}%
9161     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
9162     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%
9163     \protect\glxtrfullsep{\the\glslabeltok}%
9164     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
9165     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
9166     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
9167   }%

```

Unset the regular attribute if it has been set.

```

9168   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9169     \glsasattribute{\the\glslabeltok}{regular}%
9170     {%
9171       \glssetattribute{\the\glslabeltok}{regular}{false}%
9172     }%
9173   }%
9174 }%
9175 }%

```

```

9176 {%
9177   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
9178 }

```

`\glstrshorthyphen`
 $\glstrshorthyphen\langle short \rangle\{\langle label \rangle\}\{\langle insert \rangle\}$

Used by short-hyphen-postlong-hyphen. The $\langle insert \rangle$ is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```

9179 \newcommand*\glstrshorthyphen}[3]{%
    Grouping is needed to localise the redefinitions.
9180 {%
9181   \glstrifhyphenstart{#3}\def\glstrwordsep{-}\}%
9182   \glsfirstabbrvhyphenfont{#1}%
9183 }%
9184 }

```

`\glstrposthyphenlong`
 $\glstrposthyphenlong\{\langle label \rangle\}\{\langle insert \rangle\}$

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glstrshorthyphenlong` but omits the $\langle short \rangle$ part. This always uses the singular long form.

```

9185 \newcommand*\glstrposthyphenlong}[2]{%
9186 {%
9187   \glstrifhyphenstart{#2}\def\glstrwordsep{-}\}%
9188   \ifglstrinsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
9189   \glstrfullsep{#1}%
9190   \glstrparen
9191   {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglstrinsertinside{#2}\fi}%
9192   \ifglstrinsertinside\else{#2}\fi
9193 }%
9194 }%
9195 }

```

`postlong-hyphen` Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

9196 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
9197 {%
9198   \renewcommand*\CustomAbbreviationFields{%
9199     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
9200     sort={\the\glsshorttok},
9201     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
9202     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
9203     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%

```



```

9204     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
9205 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9206   \csdef{glsxtrpostlink\glscategorylabel}{%
9207     \glsxtrifwasfirstuse
9208     {%
9209       \glsxtrposthyphenlong{\glslabel}{\glsinsert}%
9210     }%
9211     {%

```

Put the insertion into the post-link:

```

9212       \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
9213     }%
9214   }%
9215   \glsattribute{\the\glslabeltok}{regular}%
9216   {%
9217     \glssetattribute{\the\glslabeltok}{regular}{false}%
9218   }%
9219   {}%
9220 }%
9221 }%
9222 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9223 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
9224 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9225 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9226 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9227 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

9228 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9229   \glsabbrvfont{\glsaccessshort{##1}}}%
9230 }%
9231 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9232   \glsabbrvfont{\glsaccessshortpl{##1}}}%
9233 }%
9234 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9235   \glsabbrvfont{\Glsaccessshort{##1}}}%
9236 }%
9237 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9238   \glsabbrvfont{\Glsaccessshortpl{##1}}}%
9239 }%

```

First use full form:

```

9240 \renewcommand*{\glsxtrfullformat}[2]{%
9241   \glsxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
9242 }%
9243 \renewcommand*{\glsxtrfullplformat}[2]{%
9244   \glsxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
9245 }%
9246 \renewcommand*{\Glsxtrfullformat}[2]{%

```

```

9247 \glstrshortshyphen{\Glsaccessshort{##1}}{##1}{##2}%
9248 }%
9249 \renewcommand*{\Glsxtrfullplformat}[2]{%
9250 \glstrshortshyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
9251 }%

```

In-line format. Commands like `\glstrfull` set `\glinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

9252 \renewcommand*{\glstrinlinefullformat}[2]{%
9253 \glfirstabbrvhyphenfont{\Glsaccessshort{##1}%
9254 \ifglstrinsertinside{##2}\fi}%
9255 \ifglstrinsertinside \else{##2}\fi
9256 }%
9257 \renewcommand*{\glstrinlinefullplformat}[2]{%
9258 \glfirstabbrvhyphenfont{\Glsaccessshortpl{##1}%
9259 \ifglstrinsertinside{##2}\fi}%
9260 \ifglstrinsertinside \else{##2}\fi
9261 }%
9262 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9263 \glfirstabbrvhyphenfont{\Glsaccessshort{##1}%
9264 \ifglstrinsertinside{##2}\fi}%
9265 \ifglstrinsertinside \else{##2}\fi
9266 }%
9267 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9268 \glfirstabbrvhyphenfont{\Glsaccessshortpl{##1}%
9269 \ifglstrinsertinside{##2}\fi}%
9270 \ifglstrinsertinside \else{##2}\fi
9271 }%
9272 }

```

`ong-hyphen-desc` Like `short-hyphen-postlong-hyphen` but the description must be supplied by the user.

```

9273 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
9274 {%
9275 \renewcommand*{\CustomAbbreviationFields}{%
9276 name={\glstrshortlongdescname},
9277 sort={\glstrshortlongdescsort},%
9278 first={\protect\glfirstabbrvhyphenfont{\the\glsshorttok}},%
9279 firstplural={\protect\glfirstabbrvhyphenfont{\the\glsshortpltok}},%
9280 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
9281 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
9282 }%
9283 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9284 \csdef{glstrpostlink\glscategorylabel}{%
9285 \glstrifwasfirstuse
9286 {%
9287 \glstrpostshyphenlong{\glslabel}{\glinsert}%
9288 }%
9289 }%

```

Put the insertion into the post-link:

```

9290     \glstrposthyphensubsequent{\glslabel}{\glsinsert}%
9291   }%
9292 }%
9293 \glsasattribute{\the\glslabeltok}{regular}%
9294 {%
9295   \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9296 }%
9297 {}%
9298 }%
9299 }%
9300 {%
9301   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
9302 }

```

1.6.8 Predefined Styles (No Short on First Use)

These styles show only the long form on first use and only the short form on subsequent use.

lsabbrvonlyfont

```

9303 \newcommand*{\glsabbrvonlyfont}{\glsabbrvdefaultfont}%

```

stabbrvonlyfont

```

9304 \newcommand*{\glsfirstabbrvonlyfont}{\glsabbrvonlyfont}%

```

glslongonlyfont

```

9305 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%

```

rstlongonlyfont

```

9306 \newcommand*{\glsfirstlongonlyfont}{\glslongonlyfont}%

```

The default short form suffix:

lsxtronlysuffix

```

9307 \newcommand*{\glsxtronlysuffix}{\glsxtrabbrvpluralsuffix}

```

only-short-only

```

9308 \newabbreviationstyle{long-only-short-only}%
9309 {%
9310   \renewcommand*{\CustomAbbreviationFields}{%
9311     name={\protect\glsabbrvonlyfont{\the\glsshorttok}},
9312     sort={\the\glsshorttok},
9313     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
9314     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
9315     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
9316     description={\protect\glslongonlyfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

9317 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9318   \glsasattribute{\the\glslabeltok}{regular}%

```

```

9319   {%
9320     \glssetattribute{\the\glslabeltok}{regular}{false}%
9321   }%
9322   {}%
9323 }%
9324 }%
9325 {%
9326 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtronlysuffix}%
9327 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvonlyfont{##1}}%
9328 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
9329 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
9330 \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%

```

The first use full form doesn't show the short form.

```

9331 \renewcommand*{\glsxtrfullformat}[2]{%
9332   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9333   \ifglsxtrininsertinside\else##2\fi
9334 }%
9335 \renewcommand*{\glsxtrfullplformat}[2]{%
9336   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9337   \ifglsxtrininsertinside\else##2\fi
9338 }%
9339 \renewcommand*{\Glsxtrfullformat}[2]{%
9340   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9341   \ifglsxtrininsertinside\else##2\fi
9342 }%
9343 \renewcommand*{\Glsxtrfullplformat}[2]{%
9344   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9345   \ifglsxtrininsertinside\else##2\fi
9346 }%

```

The inline full form does show the short form.

```

9347 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9348   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9349   \ifglsxtrininsertinside\else##2\fi
9350   \glsxtrfullsep{##1}%
9351   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
9352 }%
9353 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9354   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9355   \ifglsxtrininsertinside\else##2\fi
9356   \glsxtrfullsep{##1}%
9357   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
9358 }%
9359 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9360   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9361   \ifglsxtrininsertinside\else##2\fi
9362   \glsxtrfullsep{##1}%
9363   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
9364 }%

```

```

9365 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9366   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9367   \ifglxtrinsertinside\else##2\fi
9368   \glxtrfullsep{##1}%
9369   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
9370 }%
9371 }

```

xtronlydescsort

```

9372 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

9373 \newcommand*{\glxtronlydescname}{%
9374   \protect\glslongfont{\the\glslongtok}%
9375 }

```

short-only-desc

```

9376 \newabbreviationstyle{long-only-short-only-desc}%
9377 {%
9378   \renewcommand*{\CustomAbbreviationFields}{%
9379     name={\glxtronlydescname},
9380     sort={\glxtronlydescsort},%
9381     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
9382     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
9383     text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%
9384     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}}}%
9385 }%

```

Unset the regular attribute if it has been set.

```

9386 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9387   \glshasattribute{\the\glslabeltok}{regular}%
9388   {%
9389     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9390   }%
9391   {}%
9392 }%
9393 }%
9394 {%
9395   \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
9396 }

```

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:

```
9397 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
9398 \renewcommand*{\markright}[1]{%
```

```
9399 \glsxtrmarkhook
```

```
9400 \@glsxtr@org@markright{\@glsxtrinmark#1\@glsxtrnotinmark}%
```

```
9401 \glsxtrrestoremarkhook
```

```
9402 }
```

`\markboth` Save original definition:

```
9403 \let\@glsxtr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
9404 \renewcommand*{\markboth}[2]{%
```

```
9405 \glsxtrmarkhook
```

```
9406 \@glsxtr@org@markboth
```

```
9407 {\@glsxtrinmark#1\@glsxtrnotinmark}%
```

```
9408 {\@glsxtrinmark#2\@glsxtrnotinmark}%
```

```
9409 \glsxtrrestoremarkhook
```

```
9410 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

`\glsxtrRevertMarks`

```
9411 \newcommand*{\glsxtrRevertMarks}{%
```

```
9412 \let\markright\@glsxtr@org@markright
```

```
9413 \let\markboth\@glsxtr@org@markboth
```

```
9414 }
```

`\glsxtrifinmark`

```
9415 \newcommand*{\glsxtrifinmark}[2]{#2}
```

\@glxtrinmark

```
9416 \newrobustcmd*{\@glxtrinmark}{%
9417   \let\glxtrifinmark\@firstoftwo
9418 }
```

glxstrnotinmark

```
9419 \newrobustcmd*{\@glxstrnotinmark}{%
9420   \let\glxtrifinmark\@secondoftwo
9421 }
```

\glxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```
9422 \newcommand*{\glxtrmarkhook}{%
```

 Save current definitions:

```
9423   \let\@glxtr@org@MakeUppercase\MakeUppercase
9424   \let\@glxtr@org@glxstrtitleshort\glxstrtitleshort
9425   \let\@glxtr@org@glxstrtitleshortpl\glxstrtitleshortpl
9426   \let\@glxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
9427   \let\@glxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
9428   \let\@glxtr@org@glxstrtitletext\glxstrtitletext
9429   \let\@glxtr@org@Glsxtrtitletext\Glsxtrtitletext
9430   \let\@glxtr@org@glxstrtitleplural\glxstrtitleplural
9431   \let\@glxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
9432   \let\@glxtr@org@glxstrtitlefirst\glxstrtitlefirst
9433   \let\@glxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
9434   \let\@glxtr@org@glxstrtitlefirstplural\glxstrtitlefirstplural
9435   \let\@glxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
9436   \let\@glxtr@org@glxstrtitlelong\glxstrtitlelong
9437   \let\@glxtr@org@glxstrtitlelongpl\glxstrtitlelongpl
9438   \let\@glxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
9439   \let\@glxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
9440   \let\@glxtr@org@glxstrtitlefull\glxstrtitlefull
9441   \let\@glxtr@org@glxstrtitlefullpl\glxstrtitlefullpl
9442   \let\@glxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
9443   \let\@glxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl
```

 New definitions

```
9444   \let\glxtrifinmark\@firstoftwo
9445   \let\MakeUppercase\MakeTextUppercase
9446   \let\glxstrtitleshort\glxstrheadshort
9447   \let\glxstrtitleshortpl\glxstrheadshortpl
9448   \let\Glsxtrtitleshort\Glsxtrheadshort
9449   \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
9450   \let\glxstrtitletext\glxstrheadtext
9451   \let\Glsxtrtitletext\Glsxtrheadtext
9452   \let\glxstrtitleplural\glxstrheadplural
9453   \let\Glsxtrtitleplural\Glsxtrheadplural
9454   \let\glxstrtitlefirst\glxstrheadfirst
9455   \let\Glsxtrtitlefirst\Glsxtrheadfirst
```

```

9456 \let\glxstrtitlefirstplural\glxstrheadfirstplural
9457 \let\Glsxstrtitlefirstplural\Glsxtrheadfirstplural
9458 \let\glxstrtitlelong\glxstrheadlong
9459 \let\glxstrtitlelongpl\glxstrheadlongpl
9460 \let\Glsxstrtitlelong\Glsxtrheadlong
9461 \let\Glsxstrtitlelongpl\Glsxtrheadlongpl
9462 \let\glxstrtitlefull\glxstrheadfull
9463 \let\glxstrtitlefullpl\glxstrheadfullpl
9464 \let\Glsxstrtitlefull\Glsxtrheadfull
9465 \let\Glsxstrtitlefullpl\Glsxtrheadfullpl
9466 }

```

`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.)

```

9467 \newcommand*{\glxstrrestoremarkhook}{%
9468   \let\glxstrifinmark\@secondoftwo
9469   \let\MakeUppercase\@glxstr@org@MakeUppercase
9470   \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
9471   \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
9472   \let\Glsxstrtitleshort\@glxstr@org@Glsxtrtitleshort
9473   \let\Glsxstrtitleshortpl\@glxstr@org@Glsxtrtitleshortpl
9474   \let\glxstrtitletext\@glxstr@org@glxstrtitletext
9475   \let\Glsxstrtitletext\@glxstr@org@Glsxtrtitletext
9476   \let\glxstrtitleplural\@glxstr@org@glxstrtitleplural
9477   \let\Glsxstrtitleplural\@glxstr@org@Glsxtrtitleplural
9478   \let\glxstrtitlefirst\@glxstr@org@glxstrtitlefirst
9479   \let\Glsxstrtitlefirst\@glxstr@org@Glsxtrtitlefirst
9480   \let\glxstrtitlefirstplural\@glxstr@org@glxstrtitlefirstplural
9481   \let\Glsxstrtitlefirstplural\@glxstr@org@Glsxtrtitlefirstplural
9482   \let\glxstrtitlelong\@glxstr@org@glxstrtitlelong
9483   \let\glxstrtitlelongpl\@glxstr@org@glxstrtitlelongpl
9484   \let\Glsxstrtitlelong\@glxstr@org@Glsxtrtitlelong
9485   \let\Glsxstrtitlelongpl\@glxstr@org@Glsxtrtitlelongpl
9486   \let\glxstrtitlefull\@glxstr@org@glxstrtitlefull
9487   \let\glxstrtitlefullpl\@glxstr@org@glxstrtitlefullpl
9488   \let\Glsxstrtitlefull\@glxstr@org@Glsxtrtitlefull
9489   \let\Glsxstrtitlefullpl\@glxstr@org@Glsxtrtitlefullpl
9490 }

```

Instead of using one document-wide conditional, use `headuc` attribute to determine whether or not to use the all upper case form.

`glxstrheadshort` Command used to display short form in the page header.

```

9491 \newcommand*{\glxstrheadshort}[1]{%
9492   \protect\NoCaseChange
9493   {%
9494     \gl@ifattribute{#1}{headuc}{true}%

```



```

9495   {%
9496     \GLSxtrshort[noindex,hyper=false]{#1}[]%
9497   }%
9498   {%
9499     \glsxtrshort[noindex,hyper=false]{#1}[]%
9500   }%
9501 }%
9502 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

9503 \newrobustcmd*{\glsxtrtitleshort}[1]{%
9504   \glsxtrshort[noindex,hyper=false]{#1}[]%
9505 }

```

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.

```

9506 \newcommand*{\glsxtrheadshortpl}[1]{%
9507   \protect\NoCaseChange
9508   {%
9509     \glsifattribute{#1}{headuc}{true}%
9510     {%
9511       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
9512     }%
9513     {%
9514       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
9515     }%
9516   }%
9517 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```

9518 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
9519   \glsxtrshortpl[noindex,hyper=false]{#1}[]%
9520 }

```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```

9521 \newcommand*{\Glsxtrheadshort}[1]{%
9522   \protect\NoCaseChange
9523   {%
9524     \glsifattribute{#1}{headuc}{true}%
9525     {%
9526       \GLSxtrshort[noindex,hyper=false]{#1}[]%
9527     }%
9528     {%
9529       \Glsxtrshort[noindex,hyper=false]{#1}[]%
9530     }%
9531   }%
9532 }

```

`\lsxtrtitleshort` Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
9533 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
9534   \Glsxtrshort[noindex,hyper=false]{#1}[]%
9535 }
```

`\sxttheadshortpl` Command used to display plural short form in the page header with the first letter converted to upper case.

```
9536 \newcommand*{\Glsxtheadshortpl}[1]{%
9537   \protect\NoCaseChange
9538   {%
9539     \glsifattribute{#1}{headuc}{true}%
9540     {%
9541       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
9542     }%
9543     {%
9544       \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
9545     }%
9546   }%
9547 }
```

`\xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
9548 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
9549   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
9550 }
```

`\glsxtheadtext` As above but for the text value.

```
9551 \newcommand*{\glsxtheadtext}[1]{%
9552   \protect\NoCaseChange
9553   {%
9554     \glsifattribute{#1}{headuc}{true}%
9555     {%
9556       \GLStext[noindex,hyper=false]{#1}[]%
9557     }%
9558     {%
9559       \glstext[noindex,hyper=false]{#1}[]%
9560     }%
9561   }%
9562 }
```

`\glsxtrtitletext` Command to display text value in section title and table of contents.

```
9563 \newrobustcmd*{\glsxtrtitletext}[1]{%
9564   \glstext[noindex,hyper=false]{#1}[]%
9565 }
```

`\Glsxtheadtext` First letter converted to upper case

```
9566 \newcommand*{\Glsxtheadtext}[1]{%
```

```

9567 \protect\NoCaseChange
9568 {%
9569   \glsifattribute{#1}{headuc}{true}%
9570   {%
9571     \GLStext[noindex,hyper=false]{#1}[]%
9572   }%
9573   {%
9574     \GLstext[noindex,hyper=false]{#1}[]%
9575   }%
9576 }%
9577 }

```

Glsxtrtitletext Command to display text value in section title and table of contents with the first letter changed to upper case.

```

9578 \newrobustcmd*{\Glsxtrtitletext}[1]{%
9579   \GLstext[noindex,hyper=false]{#1}[]%
9580 }

```

lgsxtrheadplural As above but for the plural value.

```

9581 \newcommand*{\lgsxtrheadplural}[1]{%
9582   \protect\NoCaseChange
9583   {%
9584     \glsifattribute{#1}{headuc}{true}%
9585     {%
9586       \GLSplural[noindex,hyper=false]{#1}[]%
9587     }%
9588     {%
9589       \glsplural[noindex,hyper=false]{#1}[]%
9590     }%
9591   }%
9592 }

```

glsxtrtitleplural Command to display plural value in section title and table of contents.

```

9593 \newrobustcmd*{\glsxtrtitleplural}[1]{%
9594   \glsplural[noindex,hyper=false]{#1}[]%
9595 }

```

lgsxtrheadplural Convert first letter to upper case.

```

9596 \newcommand*{\lgsxtrheadplural}[1]{%
9597   \protect\NoCaseChange
9598   {%
9599     \glsifattribute{#1}{headuc}{true}%
9600     {%
9601       \GLSplural[noindex,hyper=false]{#1}[]%
9602     }%
9603     {%
9604       \Glsplural[noindex,hyper=false]{#1}[]%
9605     }%
9606   }%

```

```

9607 }

sxttrtitleplural Command to display plural value in section title and table of contents with the first letter
changed to upper case.
9608 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
9609   \Glsplural[noindex,hyper=false]{#1}[]%
9610 }

glxstrheadfirst As above but for the first value.
9611 \newcommand*{\glxstrheadfirst}[1]{%
9612   \protect\NoCaseChange
9613   {%
9614     \glsifattribute{#1}{headuc}{true}%
9615     {%
9616       \GLSfirst[noindex,hyper=false]{#1}[]%
9617     }%
9618     {%
9619       \glsfirst[noindex,hyper=false]{#1}[]%
9620     }%
9621   }%
9622 }

lsxtrtitlefirst Command to display first value in section title and table of contents.
9623 \newrobustcmd*{\lsxtrtitlefirst}[1]{%
9624   \glsfirst[noindex,hyper=false]{#1}[]%
9625 }

Glsxtrheadfirst First letter converted to upper case
9626 \newcommand*{\Glsxtrheadfirst}[1]{%
9627   \protect\NoCaseChange
9628   {%
9629     \glsifattribute{#1}{headuc}{true}%
9630     {%
9631       \GLSfirst[noindex,hyper=false]{#1}[]%
9632     }%
9633     {%
9634       \Glsfirst[noindex,hyper=false]{#1}[]%
9635     }%
9636   }%
9637 }

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter
changed to upper case.
9638 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
9639   \Glsfirst[noindex,hyper=false]{#1}[]%
9640 }

headfirstplural As above but for the firstplural value.

```

```

9641 \newcommand*{\glxtrheadfirstplural}[1]{%
9642   \protect\NoCaseChange
9643   {%
9644     \glusifattribute{#1}{headuc}{true}%
9645     {%
9646       \GLSfirstplural[noindex,hyper=false]{#1}[]%
9647     }%
9648     {%
9649       \GLSfirstplural[noindex,hyper=false]{#1}[]%
9650     }%
9651   }%
9652 }

```

`\titlefirstplural` Command to display firstplural value in section title and table of contents.

```

9653 \newrobustcmd*{\glxtrtitlefirstplural}[1]{%
9654   \GLSfirstplural[noindex,hyper=false]{#1}[]%
9655 }

```

`\headfirstplural` First letter converted to upper case

```

9656 \newcommand*{\Glsxtrheadfirstplural}[1]{%
9657   \protect\NoCaseChange
9658   {%
9659     \glusifattribute{#1}{headuc}{true}%
9660     {%
9661       \GLSfirstplural[noindex,hyper=false]{#1}[]%
9662     }%
9663     {%
9664       \Glsfirstplural[noindex,hyper=false]{#1}[]%
9665     }%
9666   }%
9667 }

```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```

9668 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
9669   \Glsfirstplural[noindex,hyper=false]{#1}[]%
9670 }

```

`\glxtrheadlong` Command used to display long form in the page header.

```

9671 \newcommand*{\glxtrheadlong}[1]{%
9672   \protect\NoCaseChange
9673   {%
9674     \glusifattribute{#1}{headuc}{true}%
9675     {%
9676       \GLSxtrlong[noindex,hyper=false]{#1}[]%
9677     }%
9678     {%
9679       \glxtrlong[noindex,hyper=false]{#1}[]%
9680     }%

```

```
9681 }%
9682 }
```

`\glxstrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
9683 \newrobustcmd*{\glxstrtitlelong}[1]{%
9684   \glxstrlong[noindex,hyper=false]{#1}[]%
9685 }
```

`\lsxtrheadlongpl` 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.

```
9686 \newcommand*{\lsxtrheadlongpl}[1]{%
9687   \protect\NoCaseChange
9688   {%
9689     \gl@ifattribute{#1}{headuc}{true}%
9690     {%
9691       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
9692     }%
9693     {%
9694       \glxstrlongpl[noindex,hyper=false]{#1}[]%
9695     }%
9696   }%
9697 }
```

`\sxstrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```
9698 \newrobustcmd*{\sxstrtitlelongpl}[1]{%
9699   \glxstrlongpl[noindex,hyper=false]{#1}[]%
9700 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```
9701 \newcommand*{\Glsxtrheadlong}[1]{%
9702   \protect\NoCaseChange
9703   {%
9704     \gl@ifattribute{#1}{headuc}{true}%
9705     {%
9706       \GLSxtrlong[noindex,hyper=false]{#1}[]%
9707     }%
9708     {%
9709       \Glsxtrlong[noindex,hyper=false]{#1}[]%
9710     }%
9711   }%
9712 }
```

`\Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
9713 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
9714   \Glsxtrlong[noindex,hyper=false]{#1}[]%
9715 }
```

`\lsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```

9716 \newcommand*{\Glsxtrheadlongpl}[1]{%
9717   \protect\NoCaseChange
9718   {%
9719     \glsifattribute{#1}{headuc}{true}%
9720     {%
9721       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
9722     }%
9723     {%
9724       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
9725     }%
9726   }%
9727 }

```

`\sxttrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

9728 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
9729   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
9730 }

```

`\glsxtrheadfull` Command used to display full form in the page header.

```

9731 \newcommand*{\glsxtrheadfull}[1]{%
9732   \protect\NoCaseChange
9733   {%
9734     \glsifattribute{#1}{headuc}{true}%
9735     {%
9736       \GLSxtrfull[noindex,hyper=false]{#1}[]%
9737     }%
9738     {%
9739       \glsxtrfull[noindex,hyper=false]{#1}[]%
9740     }%
9741   }%
9742 }

```

`\glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents.

```

9743 \newrobustcmd*{\glsxtrtitlefull}[1]{%
9744   \glsxtrfull[noindex,hyper=false]{#1}[]%
9745 }

```

`\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.

```

9746 \newcommand*{\glsxtrheadfullpl}[1]{%
9747   \protect\NoCaseChange
9748   {%
9749     \glsifattribute{#1}{headuc}{true}%
9750     {%

```

```

9751      \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
9752    }%
9753    {%
9754      \glxtrfullpl[noindex,hyper=false]{#1}[]%
9755    }%
9756  }%
9757 }

```

sxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents.

```

9758 \newrobustcmd*{\glxtrtitlefullpl}[1]{%
9759   \glxtrfullpl[noindex,hyper=false]{#1}[]%
9760 }

```

\Glsxtrheadfull Command used to display full form in the page header with the first letter converted to upper case.

```

9761 \newcommand*{\Glsxtrheadfull}[1]{%
9762   \protect\NoCaseChange
9763   {%
9764     \gl@ifattribute{#1}{headuc}{true}%
9765     {%
9766       \GLSxtrfull[noindex,hyper=false]{#1}[]%
9767     }%
9768     {%
9769       \Glsxtrfull[noindex,hyper=false]{#1}[]%
9770     }%
9771   }%
9772 }

```

Glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

9773 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
9774   \Glsxtrfull[noindex,hyper=false]{#1}[]%
9775 }

```

lsxtrheadfullpl Command used to display plural full form in the page header with the first letter converted to upper case.

```

9776 \newcommand*{\Glsxtrheadfullpl}[1]{%
9777   \protect\NoCaseChange
9778   {%
9779     \gl@ifattribute{#1}{headuc}{true}%
9780     {%
9781       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
9782     }%
9783     {%
9784       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
9785     }%
9786   }%
9787 }

```


`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
9788 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
9789   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
9790 }
```

`\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.

```
9791 \ifdef\texorpdfstring
9792 {
9793   \newcommand*{\glsfmtshort}[1]{%
9794     \texorpdfstring
9795       {\Glsxtrtitleshort{#1}}%
9796       {\Glsentryshort{#1}}%
9797   }
9798 }
9799 {
9800   \newcommand*{\glsfmtshort}[1]{%
9801     \Glsxtrtitleshort{#1}}
9802 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
9803 \ifdef\texorpdfstring
9804 {
9805   \newcommand*{\glsfmtshortpl}[1]{%
9806     \texorpdfstring
9807       {\Glsxtrtitleshortpl{#1}}%
9808       {\Glsentryshortpl{#1}}%
9809   }
9810 }
9811 {
9812   \newcommand*{\glsfmtshortpl}[1]{%
9813     \Glsxtrtitleshortpl{#1}}
9814 }
```

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).

```
9815 \ifdef\texorpdfstring
9816 {
9817   \newcommand*{\Glsfmtshort}[1]{%
9818     \texorpdfstring
9819       {\Glsxtrtitleshort{#1}}%
9820       {\Glsentryshort{#1}}%
9821   }
9822 }
```

```

9823 {
9824   \newcommand*{\Glsfmtshort}[1]{%
9825     \Glsxtrtitleshort{#1}}
9826 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

9827 \ifdef\teorpdfstring
9828 {
9829   \newcommand*{\Glsfmtshortpl}[1]{%
9830     \teorpdfstring
9831     {\Glsxtrtitleshortpl{#1}}%
9832     {\glstryshortpl{#1}}%
9833   }
9834 }
9835 {
9836   \newcommand*{\Glsfmtshortpl}[1]{%
9837     \Glsxtrtitleshortpl{#1}}
9838 }

```

`\glsfmttext` As above but for the text value.

```

9839 \ifdef\teorpdfstring
9840 {
9841   \newcommand*{\glsfmttext}[1]{%
9842     \teorpdfstring
9843     {\glxtrtitletext{#1}}%
9844     {\glstrytext{#1}}%
9845   }
9846 }
9847 {
9848   \newcommand*{\glsfmttext}[1]{%
9849     \glxtrtitletext{#1}}
9850 }

```

`\Glsfmttext` First letter converted to upper case.

```

9851 \ifdef\teorpdfstring
9852 {
9853   \newcommand*{\Glsfmttext}[1]{%
9854     \teorpdfstring
9855     {\Glsxtrtitletext{#1}}%
9856     {\glstrytext{#1}}%
9857   }
9858 }
9859 {
9860   \newcommand*{\Glsfmttext}[1]{%
9861     \Glsxtrtitletext{#1}}
9862 }

```

`\glsfmtplural` As above but for the plural value.

```

9863 \ifdef\teorpdfstring

```

```

9864 {
9865   \newcommand*{\glsfmtplural}[1]{%
9866     \texorpdfstring
9867     {\glsxtrtitleplural{#1}}%
9868     {\glsentryplural{#1}}%
9869   }
9870 }
9871 {
9872   \newcommand*{\glsfmtplural}[1]{%
9873     \glsxtrtitleplural{#1}}
9874 }

```

`\Glsfmtplural` First letter converted to upper case.

```

9875 \ifdef\texorpdfstring
9876 {
9877   \newcommand*{\Glsfmtplural}[1]{%
9878     \texorpdfstring
9879     {\Glsxtrtitleplural{#1}}%
9880     {\glsentryplural{#1}}%
9881   }
9882 }
9883 {
9884   \newcommand*{\Glsfmtplural}[1]{%
9885     \Glsxtrtitleplural{#1}}
9886 }

```

`\glsfmtfirst` As above but for the first value.

```

9887 \ifdef\texorpdfstring
9888 {
9889   \newcommand*{\glsfmtfirst}[1]{%
9890     \texorpdfstring
9891     {\glsxtrtitlefirst{#1}}%
9892     {\glsentryfirst{#1}}%
9893   }
9894 }
9895 {
9896   \newcommand*{\glsfmtfirst}[1]{%
9897     \glsxtrtitlefirst{#1}}
9898 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

9899 \ifdef\texorpdfstring
9900 {
9901   \newcommand*{\Glsfmtfirst}[1]{%
9902     \texorpdfstring
9903     {\Glsxtrtitlefirst{#1}}%
9904     {\glsentryfirst{#1}}%
9905   }
9906 }

```

```

9907 {
9908   \newcommand*{\Glsfmtfirst}[1]{%
9909     \Glsxtrtitlefirst{#1}}
9910 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

9911 \ifdef\texorpdfstring
9912 {
9913   \newcommand*{\glsfmtfirstpl}[1]{%
9914     \texorpdfstring
9915       {\Glsxtrtitlefirstplural{#1}}%
9916       {\Glsentryfirstplural{#1}}%
9917   }
9918 }
9919 {
9920   \newcommand*{\glsfmtfirstpl}[1]{%
9921     \Glsxtrtitlefirstplural{#1}}
9922 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

9923 \ifdef\texorpdfstring
9924 {
9925   \newcommand*{\Glsfmtfirstpl}[1]{%
9926     \texorpdfstring
9927       {\Glsxtrtitlefirstplural{#1}}%
9928       {\Glsentryfirstplural{#1}}%
9929   }
9930 }
9931 {
9932   \newcommand*{\Glsfmtfirstpl}[1]{%
9933     \Glsxtrtitlefirstplural{#1}}
9934 }

```

`\glsfmtlong` As above but for the long value.

```

9935 \ifdef\texorpdfstring
9936 {
9937   \newcommand*{\glsfmtlong}[1]{%
9938     \texorpdfstring
9939       {\Glsxtrtitlelong{#1}}%
9940       {\Glsentrylong{#1}}%
9941   }
9942 }
9943 {
9944   \newcommand*{\glsfmtlong}[1]{%
9945     \Glsxtrtitlelong{#1}}
9946 }

```

`\Glsfmtlong` First letter converted to upper case.

```

9947 \ifdef\texorpdfstring

```

```

9948 {
9949   \newcommand*{\Glsfmtlong}[1]{%
9950     \texorpdfstring
9951     {\Glsxtrtitlelong{#1}}%
9952     {\glsentrylong{#1}}%
9953   }
9954 }
9955 {
9956   \newcommand*{\Glsfmtlong}[1]{%
9957     \Glsxtrtitlelong{#1}}
9958 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

9959 \ifdef\texorpdfstring
9960 {
9961   \newcommand*{\glsfmtlongpl}[1]{%
9962     \texorpdfstring
9963     {\glsxtrtitlelongpl{#1}}%
9964     {\glsentrylongpl{#1}}%
9965   }
9966 }
9967 {
9968   \newcommand*{\glsfmtlongpl}[1]{%
9969     \glsxtrtitlelongpl{#1}}
9970 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

9971 \ifdef\texorpdfstring
9972 {
9973   \newcommand*{\Glsfmtlongpl}[1]{%
9974     \texorpdfstring
9975     {\Glsxtrtitlelongpl{#1}}%
9976     {\glsentrylongpl{#1}}%
9977   }
9978 }
9979 {
9980   \newcommand*{\Glsfmtlongpl}[1]{%
9981     \Glsxtrtitlelongpl{#1}}
9982 }

```

`\glsfmtfull` In-line full format.

```

9983 \ifdef\texorpdfstring
9984 {
9985   \newcommand*{\glsfmtfull}[1]{%
9986     \texorpdfstring
9987     {\glsxtrtitlefull{#1}}%
9988     {\glsxtrinlinefullformat{#1}{}}%
9989   }
9990 }

```

```

9991 {
9992   \newcommand*{\glsfmtfull}[1]{%
9993     \glsxtrtitlefull{#1}}
9994 }

```

`\Glsfmtfull` First letter converted to upper case.

```

9995 \ifdef\teorpdfstring
9996 {
9997   \newcommand*{\Glsfmtfull}[1]{%
9998     \teorpdfstring
9999     {\Glsxtrtitlefull{#1}}%
10000     {\Glsxtrinlinefullformat{#1}-{}}%
10001   }
10002 }
10003 {
10004   \newcommand*{\Glsfmtfull}[1]{%
10005     \Glsxtrtitlefull{#1}}
10006 }

```

`\glsfmtfullpl` In-line full plural format.

```

10007 \ifdef\teorpdfstring
10008 {
10009   \newcommand*{\glsfmtfullpl}[1]{%
10010     \teorpdfstring
10011     {\glsxtrtitlefullpl{#1}}%
10012     {\glsxtrinlinefullplformat{#1}-{}}%
10013   }
10014 }
10015 {
10016   \newcommand*{\glsfmtfullpl}[1]{%
10017     \glsxtrtitlefullpl{#1}}
10018 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

10019 \ifdef\teorpdfstring
10020 {
10021   \newcommand*{\Glsfmtfullpl}[1]{%
10022     \teorpdfstring
10023     {\Glsxtrtitlefullpl{#1}}%
10024     {\Glsxtrinlinefullplformat{#1}-{}}%
10025   }
10026 }
10027 {
10028   \newcommand*{\Glsfmtfullpl}[1]{%
10029     \Glsxtrtitlefullpl{#1}}
10030 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

sariesExtraLang

```
10031 \newcommand*{\RequireGlossariesExtraLang}[1]{%
10032   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
10033 }
```

sariesExtraLang

```
10034 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
10035   \ProvidesFile{glossariesxtr-#1.ldf}%
10036 }
```

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.)

```
10037 \@ifpackageloaded{tracklang}
10038 {%
10039   \AnyTrackedLanguages
10040   {%
10041     \ForEachTrackedDialect{\this@dialect}{%
10042       \IfTrackedLanguageFileExists{\this@dialect}%
10043       {glossariesxtr-}% prefix
10044       {.ldf}%
10045       {%
10046         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
10047       }%
10048     }%
10049   }%
10050 }%
10051 }%
10052 {}%
10053 }
10054 {}
```

Load `glossaries-extra-stylemods` if required.

```
10055 \@glsxtr@redefstyles
```

and set the style:

```
10056 \@glsxtr@do@style
```

2 Style Adjustments (glossaries-extra-stylemods.sty)

This package adjusts the predefined styles so that they include the post description hook. Also, some other minor adjustments may be made to make existing styles more flexible.

2.1 Package Initialisation

First identify package:

```
10057 \NeedsTeXFormat{LaTeX2e}
10058 \ProvidesPackage{glossaries-extra-stylemods}[2017/08/09 v1.17 (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`

```
10059 \newcommand*{\@glxtr@loadstyles}{%
10060 \DeclareOption*{%
10061   \IfFileExists{glossary-\CurrentOption.sty}%
10062     {\eappto\@glxtr@loadstyles{%
10063       \noexpand\RequirePackage{glossary-\CurrentOption}}}%
10064     {\PackageError{glossaries-extra-styles}%
10065       {Unknown option '\CurrentOption'}{}}
10066 }
```

Process the package options:

```
10067 \ProcessOptions
```

Load the required packages:

```
10068 \@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`

```
10069 \providecommand{\renewglossarystyle}[2]{%
10070   \ifcsundef{@glstyle@#1}%
10071     {%
10072       \PackageError{glossaries}{Glossary style '#1' isn't already defined}{}%

```



```

10073 }%
10074 {%
10075     \csdef{@glsstyle@#1}{#2}%
10076 }%
10077 }

```

2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying.

```

10078 \ifcsdef{@glsstyle@listdotted}
10079 {%
10080     \renewglossarystyle{listdotted}{%
10081         \setglossarystyle{list}%
10082         \renewcommand*{\glossentry}[2]{%
10083             \item[]\makebox[\glslistdottedwidth][l]{%
10084                 \glstryitem{##1}%
10085                 \glstarget{##1}{\glossentryname{##1}}%
10086                 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
10087                 \glossentrydesc{##1}\glspostdescription}%
10088         \renewcommand*{\subglossentry}[3]{%
10089             \item[]\makebox[\glslistdottedwidth][l]{%
10090                 \glssubentryitem{##2}%
10091                 \glstarget{##2}{\glossentryname{##2}}%
10092                 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
10093                 \glossentrydesc{##2}\glspostdescription}%
10094     }
10095 }
10096 {}

```

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.

```

10097 \ifcsdef{@glsstyle@long3col}
10098 {%
10099     \renewglossarystyle{long3col}{%
10100         \renewenvironment{theglossary}%
10101             {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
10102             {\end{longtable}}%
10103         \renewcommand*{\glossaryheader}{}%
10104         \renewcommand*{\glsgroupheading}[1]{%
10105             \renewcommand{\glossentry}[2]{%
10106                 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10107                 \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline

```

```

10108 }%
10109 \renewcommand{\subglossentry}[3]{%
10110     &
10111     \glssubentryitem{##2}%
10112     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10113     ##3\tabularnewline
10114 }%
10115 \renewcommand*{\glsgroupskip}{%
10116     \ifglsgnোগroupskip\else & &\tabularnewline\fi}%
10117 }
10118 }
10119 {}

```

Four column style:

```

10120 \ifcsdef{@glsstyle@long4col}
10121 {%
10122     \renewglossarystyle{long4col}{%
10123         \renewenvironment{theglossary}%
10124             {\begin{longtable}{llll}}%
10125             {\end{longtable}}%
10126         \renewcommand*{\glossaryheader}{}%
10127         \renewcommand*{\glsgroupheading}[1]{}%
10128         \renewcommand{\glossentry}[2]{%
10129             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10130             \glossentrydesc{##1}\glspostdescription &
10131             \glossentrysymbol{##1} &
10132             ##2\tabularnewline
10133         }%
10134         \renewcommand{\subglossentry}[3]{%
10135             &
10136             \glssubentryitem{##2}%
10137             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10138             \glossentrysymbol{##2} & ##3\tabularnewline
10139         }%
10140         \renewcommand*{\glsgroupskip}{%
10141             \ifglsgnোগroupskip\else & & &\tabularnewline\fi}%
10142     }
10143 }
10144 {}

```

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.

```

10145 \ifcsdef{@glsstyle@longragged3col}
10146 {%
10147     \renewglossarystyle{longragged3col}{%

```

```

10148 \renewenvironment{theglossary}%
10149     {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}%
10150      >{\raggedright}p{\glspagelistwidth}}}%
10151     {\end{longtable}}}%
10152 \renewcommand*{\glossaryheader}{}%
10153 \renewcommand*{\glsgroupheading}[1]{}%
10154 \renewcommand{\glossentry}[2]{%
10155     \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10156     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
10157 }%
10158 \renewcommand{\subglossentry}[3]{%
10159     &
10160     \glssubentryitem{##2}%
10161     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10162     ##3\tabularnewline
10163 }%
10164 \renewcommand*{\glsgroupskip}{%
10165     \ifglsgnogroupskip\else & \tabularnewline\fi}%
10166 }
10167 }
10168 {}

```

Four column style:

```

10169 \ifcsdef{@glstyle@altlongragged4col}
10170 {%
10171     \renewglossarystyle{altlongragged4col}{%
10172         \renewenvironment{theglossary}%
10173             {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}l%
10174              >{\raggedright}p{\glspagelistwidth}}}%
10175             {\end{longtable}}}%
10176         \renewcommand*{\glossaryheader}{}%
10177         \renewcommand*{\glsgroupheading}[1]{}%
10178         \renewcommand{\glossentry}[2]{%
10179             \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10180             \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
10181             ##2\tabularnewline
10182         }%
10183         \renewcommand{\subglossentry}[3]{%
10184             &
10185             \glssubentryitem{##2}%
10186             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10187             \glossentrysymbol{##2} & ##3\tabularnewline
10188         }%
10189         \renewcommand*{\glsgroupskip}{%
10190             \ifglsgnogroupskip\else & \tabularnewline\fi}%
10191     }
10192 }
10193 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment, but not the two column styles.

```
10194 \ifcsdef{@glsstyle@super3col}
10195 {%
10196   \renewglossarystyle{super3col}{%
10197     \renewenvironment{theglossary}%
10198       {\tablehead{}}\tabletail{}}%
10199     \begin{supertabular}{\lp{@glsdescwidth}p{@glspagelistwidth}}}%
10200     {\end{supertabular}}}%
10201   \renewcommand*{\glossaryheader}{}%
10202   \renewcommand*{\glsgroupheading}[1]{}%
10203   \renewcommand{\glossentry}[2]{%
10204     \glstryitem{##1}\glstarget{##1}{\glsentryname{##1}} &
10205     \glsentrydesc{##1}\glspostdescription & ##2\tabularnewline
10206   }%
10207   \renewcommand{\subglossentry}[3]{%
10208     &
10209     \glssubentryitem{##2}%
10210     \glstarget{##2}{\strut}\glsentrydesc{##2}\glspostdescription &
10211     ##3\tabularnewline
10212   }%
10213   \renewcommand*{\glsgroupskip}{%
10214     \ifglsgnpgroupskip\else & &\tabularnewline\fi}%
10215 }
10216 }
10217 {}
```

Four column styles:

```
10218 \ifcsdef{@glsstyle@super4col}
10219 {%
10220   \renewglossarystyle{super4col}{%
10221     \renewenvironment{theglossary}%
10222       {\tablehead{}}\tabletail{}}%
10223     \begin{supertabular}{\llll}}}%
10224     \end{supertabular}}}%
10225   \renewcommand*{\glossaryheader}{}%
10226   \renewcommand*{\glsgroupheading}[1]{}%
10227   \renewcommand{\glossentry}[2]{%
10228     \glstryitem{##1}\glstarget{##1}{\glsentryname{##1}} &
10229     \glsentrydesc{##1}\glspostdescription &
10230     \glsentrysymbol{##1} & ##2\tabularnewline
10231   }%
10232   \renewcommand{\subglossentry}[3]{%
10233     &
10234     \glssubentryitem{##2}%
10235     \glstarget{##2}{\strut}\glsentrydesc{##2}\glspostdescription &
10236     \glsentrysymbol{##2} & ##3\tabularnewline
10237   }%
10238   \renewcommand*{\glsgroupskip}{%
```

```

10239 \ifglsnogroupskip\else & &\tabularnewline\fi}%
10240 }
10241 }
10242 {}

```

2.6 Super Ragged Styles

The three and four column styles require adjustment, but not the two column styles.

```

10243 \ifcsdef{@glsstyle@superragged3col}
10244 {%
10245 \renewglossarystyle{superragged3col}{%
10246 \renewenvironment{theglossary}%
10247 {\tablehead{}\tabletail}%
10248 \begin{supertabular}{1>\raggedright}p{\glsdescwidth}%
10249 >\raggedright}p{\glspagelistwidth}}}%
10250 {\end{supertabular}}}%
10251 \renewcommand*{\glossaryheader}{}%
10252 \renewcommand*{\glsgroupheading}[1]{}%
10253 \renewcommand{\glossentry}[2]{%
10254 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10255 \glossentrydesc{##1}\glspostdescription &
10256 ##2\tabularnewline
10257 }%
10258 \renewcommand{\subglossentry}[3]{%
10259 &
10260 \glssubentryitem{##2}%
10261 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10262 ##3\tabularnewline
10263 }%
10264 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
10265 &\tabularnewline\fi}%
10266 }
10267 }
10268 {}

```

Four columns:

```

10269 \ifcsdef{@glsstyle@altsuperragged4col}
10270 {%
10271 \renewglossarystyle{altsuperragged4col}{%
10272 \renewenvironment{theglossary}%
10273 {\tablehead{}\tabletail}%
10274 \begin{supertabular}{1>\raggedright}p{\glsdescwidth}l%
10275 >\raggedright}p{\glspagelistwidth}}}%
10276 {\end{supertabular}}}%
10277 \renewcommand*{\glossaryheader}{}%
10278 \renewcommand{\glossentry}[2]{%
10279 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10280 \glossentrydesc{##1}\glspostdescription &
10281 \glossentrysymbol{##1} & ##2\tabularnewline

```

```

10282 }%
10283 \renewcommand{\subglossentry}[3]{%
10284     &
10285     \glssubentryitem{##2}%
10286     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10287     \glossentrysymbol{##2} & ##3\tabularnewline
10288 }%
10289 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & &
10290     &\tabularnewline\fi}%
10291 }
10292 }
10293 {}

```

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`.

```

10294 \ifdef{\@glsstyle@inline}
10295 {%
10296     \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
10297     Just use \glsxtrpostdescription instead of \glspostdescription.
10298     \renewcommand*{\glsinlinedescformat}[3]{%
10299         \space#1\glsxtrpostdescription}
10300     \renewcommand*{\glsinlinesubdescformat}[3]{%
10301         #1\glsxtrpostdescription}
10302 }

```

2.8 Tree Styles

The `alttree` style is redefined to make it easier to made minor adjustments.

```

10303 \ifdef{\@glsstyle@alttree}
10304 {%

```

Only redefine this style if it's already been defined.

mbolDescLocation

```
\glxtraltrtreeSymbolDescLocation{\langle label \rangle}{\langle location list \rangle}
```

Layout the symbol, description and location for top-level entries.

```

10305 \newcommand{\glxtraltrtreeSymbolDescLocation}[2]{%
10306     {%
10307         \let\par\glsxtrAltTreePar

```

```

10308      \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
10309      \glossentrydesc{#1}\glspostdescription \space #2\par
10310    }%
10311  }

```

`\trAltTreeIndent` Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

10312  \newlength\glxstrAltTreeIndent

```

`\lsxtrAltTreePar` Multi-paragraph descriptions need to keep the hanging indent.

```

10313  \newcommand{\glxstrAltTreePar}{%
10314    @@par
10315    \glxstrAltTreeSetHangIndent
10316    \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
10317  }

```

`\symbolDescLocation` `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

10318  \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
10319    \glxstralttreeSymbolDescLocation{#2}{#3}%
10320  }

```

`\trtreeTopindent` The original style has to keep computing the width of the name at each entry. This register allows the style to compute it once for the top-level at the start of the glossary.

```

10321  \newlength\glxstrtreeTopindent

```

`\glxstralttreeInit` User-level initialisation for the alttree style.

```

10322  \newcommand*{\glxstralttreeInit}{%
10323    \settowidth{\glxstrtreeTopindent}{\glstreenamefmt{\glsgetwidestname\space}}%
10324    \glxstrAltTreeIndent=\parindent
10325  }

```

`\eglissetwidest` The original `\glissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

10326  \newcommand*{\eglissetwidest}[2][0]{%
10327    \protected@csedef{@glswidestname\romannumeral#1}{#2}%
10328  }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```

10329  \newcommand*{\xglissetwidest}[2][0]{%
10330    \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
10331  }

```

`\lsgetwidestname` Provide a user-level macro to obtain the widest top-level name.

```

10332  \newcommand*{\lsgetwidestname}{\@glswidestname}

```

`etwidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```

10333 \newcommand*\glsgetwidestsubname}[1]{%
10334 \ifcsundef{@glswidestname\romannumeral#1}%
10335 {\@glswidestname}%
10336 {\csuse{@glswidestname\romannumeral#1}}%
10337 }

```

`estTopLevelName` CamelCase is easier for long command names. Provide a CamelCase synonym of `\glsfindwidesttoplevelname`

```

10338 \let\glsFindWidestTopLevelName\glsfindwidesttoplevelname

```

`sedTopLevelName` Like `\glsfindwidesttoplevelname` but has an additional check that the entry has been used. Only useful if the glossaries occur at the end of the document, in which case this command should go at the start of the glossary. Alternatively, place at the end of the document and save for the next run.

```

10339 \newrobustcmd*\glsFindWidestUsedTopLevelName}[1][\@glo@types]{%
10340 \dimen@=0pt\relax
10341 \gls@tmplen=0pt\relax
10342 \foralllglossaries[#1]{\@gls@type}%
10343 {%
10344 \forglsentries[\@gls@type]{\@glo@label}%
10345 {%
10346 \ifglsused{\@glo@label}%
10347 {%
10348 \ifglshasparent{\@glo@label}%
10349 {}%
10350 {%
10351 \settowidth{\dimen@}%
10352 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
10353 \ifdim\dimen@>\gls@tmplen
10354 \gls@tmplen=\dimen@
10355 \eglssetwidest{\glsentryname{\@glo@label}}%
10356 \fi
10357 }%
10358 }%
10359 {}%
10360 }%
10361 }%
10362 }

```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

10363 \newrobustcmd*\glsFindWidestUsedAnyName}[1][\@glo@types]{%
10364 \dimen@=0pt\relax
10365 \gls@tmplen=0pt\relax
10366 \foralllglossaries[#1]{\@gls@type}%
10367 {%
10368 \forglsentries[\@gls@type]{\@glo@label}%
10369 {%

```



```

10370     \ifglused{\@glo@label}%
10371     {%
10372         \settowidth{\dimen@}%
10373         {\glstreenamefmt{\glentryname{\@glo@label}}}%
10374         \ifdim\dimen@>\gls@tmplen
10375             \gls@tmplen=\dimen@
10376             \eglssetwidest{\glentryname{\@glo@label}}%
10377         \fi
10378     }%
10379     {%
10380     }%
10381     }%
10382 }

```

FindWidestAnyName Like the above but doesn't check if the entry has been used.

```

10383 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
10384     \dimen@=0pt\relax
10385     \gls@tmplen=0pt\relax
10386     \forallglossaries[#1]{\@gls@type}%
10387     {%
10388         \forallglentries[\@gls@type]{\@glo@label}%
10389         {%
10390             \settowidth{\dimen@}%
10391             {\glstreenamefmt{\glentryname{\@glo@label}}}%
10392             \ifdim\dimen@>\gls@tmplen
10393                 \gls@tmplen=\dimen@
10394                 \eglssetwidest{\glentryname{\@glo@label}}%
10395             \fi
10396         }%
10397     }%
10398 }

```

FindUsedLevelTwo This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well.

Any entry that has a great-grandparent is ignored.

```

10399 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
10400     \dimen@=0pt\relax
10401     \dimen@i=0pt\relax
10402     \dimen@ii=0pt\relax
10403     \forallglossaries[#1]{\@gls@type}%
10404     {%
10405         \forallglentries[\@gls@type]{\@glo@label}%
10406         {%
10407             \ifglused{\@glo@label}%
10408             {%
10409                 \ifglshasparent{\@glo@label}%
10410                 {%
10411                     \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
10412                     \ifglshasparent{\@glo@parent}%
10413                 }%

```

```

10414      \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
10415      \ifglshasparent{\@glo@parent}%
10416      {%
10417      {%
10418          \settowidth{\gls@tmplen}%
10419              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10420          \ifdim\gls@tmplen>\dimen@ii
10421              \dimen@ii=\gls@tmplen
10422              \eglssetwidest[2]{\glsentryname{\@glo@label}}%
10423          \fi
10424      }%
10425      }%
10426      {%
10427          \settowidth{\gls@tmplen}%
10428              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10429          \ifdim\gls@tmplen>\dimen@i
10430              \dimen@i=\gls@tmplen
10431              \eglssetwidest[1]{\glsentryname{\@glo@label}}%
10432          \fi
10433      }%
10434      }%
10435      {%
10436          \settowidth{\gls@tmplen}%
10437              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10438          \ifdim\gls@tmplen>\dimen@
10439              \dimen@=\gls@tmplen
10440              \eglssetwidest{\glsentryname{\@glo@label}}%
10441          \fi
10442      }%
10443      }%
10444      {}%
10445      }%
10446      }%
10447      }

```

`dWidestLevelTwo` This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

10448      \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
10449          \dimen@=0pt\relax
10450          \dimen@i=0pt\relax
10451          \dimen@ii=0pt\relax
10452          \forallglossaries[#1]{\@gls@type}%
10453          {%
10454              \forglsentries[\@gls@type]{\@glo@label}%
10455              {%
10456                  \ifglshasparent{\@glo@label}%
10457                  {%
10458                      \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
10459                      \ifglshasparent{\@glo@parent}%
10460                      {%

```

```

10461      \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
10462      \ifglshasparent{\@glo@parent}%
10463      {%
10464      {%
10465          \settowidth{\gls@tmplen}%
10466              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10467          \ifdim\gls@tmplen>\dimen@ii
10468              \dimen@ii=\gls@tmplen
10469              \eglssetwidest[2]{\glsentryname{\@glo@label}}%
10470          \fi
10471      }%
10472      }%
10473      {%
10474          \settowidth{\gls@tmplen}%
10475              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10476          \ifdim\gls@tmplen>\dimen@i
10477              \dimen@i=\gls@tmplen
10478              \eglssetwidest[1]{\glsentryname{\@glo@label}}%
10479          \fi
10480      }%
10481      }%
10482      {%
10483          \settowidth{\gls@tmplen}%
10484              {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10485          \ifdim\gls@tmplen>\dimen@
10486              \dimen@=\gls@tmplen
10487              \eglssetwidest{\glsentryname{\@glo@label}}%
10488          \fi
10489      }%
10490      }%
10491      }%
10492      }

```

edAnyNameSymbol Like the `\glsFindWidestUsedAnyName` but also measures the symbol. The length of the widest symbol is stored in the second argument should be a length register.

```

10493 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
10494     \dimen@=0pt\relax
10495     \gls@tmplen=0pt\relax
10496     #2=0pt\relax
10497     \forallglossaries[#1]{\@gls@type}%
10498     {%
10499         \forglsentries[\@gls@type]{\@glo@label}%
10500         {%
10501             \ifglused{\@glo@label}%
10502             {%
10503                 \settowidth{\dimen@}%
10504                     {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10505                 \ifdim\dimen@>\gls@tmplen
10506                     \gls@tmplen=\dimen@

```

```

10507         \eglssetwidest{\glsentryname{\@glo@label}}}%
10508         \fi
10509         \settowidth{\dimen@}%
10510         {\glsentrysymbol{\@glo@label}}}%
10511         \ifdim\dimen@>#2\relax
10512             #2=\dimen@
10513         \fi
10514     }%
10515     {}%
10516 }%
10517 }%
10518 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

10519 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
10520     \dimen@=0pt\relax
10521     \gls@tmplen=0pt\relax
10522     #2=0pt\relax
10523     \forallglossaries[#1]{\@gls@type}%
10524     {%
10525         \forglsentries[\@gls@type]{\@glo@label}%
10526         {%
10527             \settowidth{\dimen@}%
10528             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
10529             \ifdim\dimen@>\gls@tmplen
10530                 \gls@tmplen=\dimen@
10531                 \eglssetwidest{\glsentryname{\@glo@label}}}%
10532             \fi
10533             \settowidth{\dimen@}%
10534             {\glsentrysymbol{\@glo@label}}}%
10535             \ifdim\dimen@>#2\relax
10536                 #2=\dimen@
10537             \fi
10538         }%
10539     }%
10540 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but also measures the location list. This requires `\glsentrynumberlist`. The length of the widest symbol is stored in the second argument should be a length register. The length of the widest location list is stored in the third argument, which should also be a length register.

```

10541 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
10542     \dimen@=0pt\relax
10543     \gls@tmplen=0pt\relax
10544     #2=0pt\relax
10545     #3=0pt\relax
10546     \forallglossaries[#1]{\@gls@type}%
10547     {%
10548         \forglsentries[\@gls@type]{\@glo@label}%

```

```

10549    {%
10550        \ifglsused{\@glo@label}%
10551    {%
10552        \settowidth{\dimen@}%
10553            {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10554        \ifdim\dimen@>\gls@tmplen
10555            \gls@tmplen=\dimen@
10556            \eglssetwidest{\glsentryname{\@glo@label}}%
10557        \fi
10558        \settowidth{\dimen@}%
10559            {\glsentrysymbol{\@glo@label}}%
10560        \ifdim\dimen@>#2\relax
10561            #2=\dimen@
10562        \fi
10563        \settowidth{\dimen@}%
10564            {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
10565        \ifdim\dimen@>#3\relax
10566            #3=\dimen@
10567        \fi
10568    }%
10569    {%
10570    }%
10571    }%
10572 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

10573 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
10574     \dimen@=0pt\relax
10575     \gls@tmplen=0pt\relax
10576     #2=0pt\relax
10577     #3=0pt\relax
10578     \forallglossaries[#1]{\@gls@type}%
10579     {%
10580         \forglsentries[\@gls@type]{\@glo@label}%
10581         {%
10582             \settowidth{\dimen@}%
10583                 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10584             \ifdim\dimen@>\gls@tmplen
10585                 \gls@tmplen=\dimen@
10586                 \eglssetwidest{\glsentryname{\@glo@label}}%
10587             \fi
10588             \settowidth{\dimen@}%
10589                 {\glsentrysymbol{\@glo@label}}%
10590             \ifdim\dimen@>#2\relax
10591                 #2=\dimen@
10592             \fi
10593             \settowidth{\dimen@}%
10594                 {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
10595             \ifdim\dimen@>#3\relax

```

```

10596         #3=\dimen@
10597     \fi
10598 }%
10599 }%
10600 }

```

AnyNameLocation Like the `\glsFindWidestUsedAnyNameSymbolLocation` but doesn't measure the symbol. The length of the widest location list is stored in the second argument, which should be a length register.

```

10601 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
10602     \dimen@=0pt\relax
10603     \gls@tmplen=0pt\relax
10604     #2=0pt\relax
10605     \forallglossaries[#1]{\@gls@type}%
10606     {%
10607         \forallglsentries[\@gls@type]{\@glo@label}%
10608         {%
10609             \ifglsused{\@glo@label}%
10610             {%
10611                 \settowidth{\dimen@}%
10612                     {\glstreenamfmt{\glsentryname{\@glo@label}}}%
10613                 \ifdim\dimen@>\gls@tmplen
10614                     \gls@tmplen=\dimen@
10615                     \eglssetwidest{\glsentryname{\@glo@label}}%
10616                 \fi
10617                 \settowidth{\dimen@}%
10618                     {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
10619                 \ifdim\dimen@>#2\relax
10620                     #2=\dimen@
10621                 \fi
10622             }%
10623         }%
10624     }%
10625 }%
10626 }

```

AnyNameLocation Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

10627 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
10628     \dimen@=0pt\relax
10629     \gls@tmplen=0pt\relax
10630     #2=0pt\relax
10631     \forallglossaries[#1]{\@gls@type}%
10632     {%
10633         \forallglsentries[\@gls@type]{\@glo@label}%
10634         {%
10635             \settowidth{\dimen@}%
10636                 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
10637             \ifdim\dimen@>\gls@tmplen
10638                 \gls@tmplen=\dimen@

```

```

10639         \eglssetwidest{\glstryname{\@glo@label}}}%
10640     \fi
10641     \settowidth{\dimen@}%
10642         {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
10643     \ifdim\dimen@>#2\relax
10644         #2=\dimen@
10645     \fi
10646 }%
10647 }%
10648 }

```

computeTreeIndent Compute the value of `\glstreeindent`. Argument is the entry label. (Ignored in default definition, but this command may be redefined to take the particular entry into account.) Note that the sub-levels modify `\glstreeindent`.

```

10649 \newcommand*\glsxtrComputeTreeIndent}[1]{%
10650     \glstreeindent=\glsxtrtreetopindent\relax
10651 }

```

computeTreeSubIndent `\glsxtrComputeTreeSubIndent{<level>}{<label>}{<register>}`

Compute the indent for the sub-entries. The first argument is the level, the second argument is the entry label and the third argument is the length register used to store the computed indent.

```

10652 \newcommand*\glsxtrComputeTreeSubIndent}[3]{%
10653     \ifcsundef{@glswidestname\romannumeral#1}%
10654     {%
10655         \settowidth{#3}{\glstreenamfmt{\@glswidestname\space}}%
10656     }%
10657     {%
10658         \settowidth{#3}{\glstreenamfmt{%
10659             \csname @glswidestname\romannumeral#1\endcsname\space}}%
10660     }%
10661 }

```

treeSetHangIndent Set `\hangindent` for top-level entries:

```

10662 \newcommand*\glsxtrAltTreeSetHangIndent{\hangindent\glstreeindent}

```

treeSubHangIndent Set `\hangindent` for sub-entries:

```

10663 \newcommand*\glsxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine `alttree`:

```

10664 \renewglossarystyle{alttree}{%
10665     \renewenvironment{theglossary}%
10666     {%
10667         \glsxtralttreeInit

```

```

10668     \def\@gls@prevlevel{-1}%
10669     \mbox{}\par}%
10670     {\par}%
10671 \renewcommand*\glossaryheader{}%
10672 \renewcommand*\glsgroupheading[1]{}%
10673 \renewcommand\glossentry[2]{%
10674     \ifnum\@gls@prevlevel=0\relax
10675     \else
10676         \glstrComputeTreeIndent{##1}%
10677     \fi
10678     \parindent\glstreeindent
10679     \glstrAltTreeSetHangIndent
10680     \makebox[Opt][r]%
10681     {%
10682         \glstreenamebox{\glstreeindent}%
10683         {%
10684             \glstryitem{##1}%
10685             \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
10686         }%
10687     }%
10688     \glstralttreeSymbolDescLocation{##1}{##2}%
10689     \def\@gls@prevlevel{0}%
10690 }
10691 \renewcommand\subglossentry[3]{%
10692     \ifnum##1=1\relax
10693         \glssubentryitem{##2}%
10694     \fi
10695     \ifnum\@gls@prevlevel=##1\relax
10696     \else
10697         \glstrComputeTreeSubIndent{##1}{##2}{\@gls@tmplen}%
10698         \ifnum\@gls@prevlevel<##1\relax
10699             \setlength\glstreeindent\@gls@tmplen
10700             \addtolength\glstreeindent\parindent
10701             \parindent\glstreeindent
10702         \else
10703             \ifnum\@gls@prevlevel=0\relax
10704                 \glstrComputeTreeIndent{##2}%
10705             \else
10706                 \glstrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
10707             \fi
10708             \addtolength\parindent{-\glstreeindent}%
10709             \setlength\glstreeindent\parindent
10710         \fi
10711     \fi
10712     \glstrAltTreeSetSubHangIndent{##1}%
10713     \makebox[Opt][r]{\glstreenamebox{\@gls@tmplen}{%
10714         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
10715     \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
10716     \def\@gls@prevlevel{##1}%

```



```

10717     }%
10718     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
10719   }
10720 }%
10721 {%

```

Assume the style isn't required if it hasn't already been defined.

```

10722 }

Reset the default style
10723 \ifx\@glossary@default@style\relax
10724 \else
10725   \setglossarystyle{\@glxtr@current@style}
10726 \fi

```

Glossary

First use The first time a glossary entry is used (from the start of the document or after a reset) with one of the following commands: `\gls`, `\Gls`, `\GLS`, `\glspl`, `\Glspl`, `\GLSpl` or `\glsdisp`. *see* **first use flag** & **first use text**

First use flag A conditional that determines whether or not the entry has been used according to the rules of **first use**.

First use text The text that is displayed on **first use**, which is governed by the first and first-plural keys of `\newglossaryentry`. (May be overridden by `\glsdisp`.)

`makeindex` An indexing application.

`xindy` An flexible indexing application with multilingual support written in Perl.

Change History

0.1 (2015-11-22)		\@Glsymbol@: added redefinition	56
General: Initial experimental release	5	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	56
\Glsfmtshort: new	273	\@Glstext@: added redefinition	52
\glsfmtshort: new	273	\@Glsuseri@: added redefinition	57
\Glsfmtshortpl: new	274	\@Glsuserii@: added redefinition	57
\glsfmtshortpl: new	273	\@Glsuseriii@: added redefinition	57
short: switched inline full form to short		\@Glsuseriv@: added redefinition	57
(long)	182	\@Glsuserv@: added redefinition	58
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	58
\@ACRlong: added redefinition	61	\@acrlong: added redefinition	61
\@ACRlongpl: added redefinition	62	\@acrlongpl: added redefinition	62
\@ACRshort: added redefinition	59	\@acrshort: added redefinition	58
\@ACRshortpl: added redefinition	60	\@acrshortpl: added redefinition	59
\@Acrlong: added redefinition	61	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	62	argument	47
\@Acrshort: added redefinition	59	\@glsdescplural@: added redefinition	55
\@Acrshortpl: added redefinition	60	\@glsfirst@: added redefinition	52
\@GLSdesc@: added redefinition	55	\@glsfirstplural@: added redefinition	53
\@GLSdescplural@: added redefinition	55	\@glsplural@: added redefinition	53
\@GLSfirst@: added redefinition	52	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	54	redefinition	56
\@GLSname@: added redefinition	54	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	53	new	109
\@GLSsymbol@: added redefinition	56	\@glsxtr@field@linkdefs: new	51
\@GLSsymbolplural@: added		\@glsxtr@insertdots: new	152
redefinition	56	\@print@glossary: added redefinition	106
\@GLStext@: added redefinition	51	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	57	\abbrvdefaultfont	157
\@GLSuserii@: added redefinition	57	\glsaccessdesc: new	121
\@GLSuseriii@: added redefinition	57	\glsaccessdescplural: new	122
\@GLSuseriv@: added redefinition	58	\glsaccessfirst: new	119
\@GLSuserv@: added redefinition	58	\glsaccessfirstplural: new	120
\@GLSuservi@: added redefinition	58	\Glsaccesslong: new	124
\@Glsdesc@: added redefinition	55	\glsaccesslong: new	124
\@Glsdescplural@: added redefinition	55	\glsaccessname: new	117
\@Glsfirst@: added redefinition	52	\glsaccessplural: new	118
\@Glsfirstplural@: added redefinition	54	\Glsaccessshort: new	123
\@Glsname@: added redefinition	54	\glsaccessshort: new	123
\@Glsplural@: added redefinition	53	\Glsaccessshortpl: new	123

\glsaccessshortpl: new	123	\cGLSpl: new	85
\glsaccesssymbol: new	120	\cGLSpl@: new	85
\glsaccesssymbolplural: new	121	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	118	new	80
\glsentryfmt: added check for short ..	46	\cGLS: new	85
\glslongpltok: new	152	\cGLSformat: new	85
\glsshortpltok: new	152	\cGLSpl: new	85
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	85
name in \setkeys	153	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	13
for plural	149	\glsenableentrycount: new	80
\GLSxtrlongpl: new	166	\glsfirstabbrvdefaultfont: new ..	157
\Glsxtrlongpl: new	166	\glsfirstlongdefaultfont: new ...	157
\glsxtrlongpl: new	165	\Glsfmtfirst: new	275
\glsxtrNoGlossaryWarning: new	17	\glsfmtfirst: new	275
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	276
new	148	\glsfmtfirstpl: new	276
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	275
new	148	\glsfmtplural: new	274
\glsxtrpostlinkendsentence: new ..	148	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	165	\Glsxtrtitleshort	273
\Glsxtrshortpl: new	164	renamed from \Glsentryfmtshort ..	273
\glsxtrshortpl: new	164	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort	273
\glslabeltok	177	renamed from \glsentryfmtshort ..	273
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	175	\Glsxtrtitleshortpl	274
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glsentryfmtshortpl	274
redefinition of \acronymtype	13	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	273
\Glsxtrshort	273	renamed from	
\glsfmtshort: changed to use		\glsentryfmtshortpl	273
\glsxtrshort	273	\Glsfmttext: new	274
\Glsfmtshortpl: changed to use		\glsfmttext: new	274
\glsxtrshortpl	274	\glshasattribute: new	129
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	129
\glsxtrshortpl	273	\glsxtremsuffix: new	215
\glsxtrifemptyglossary: new	20	\GlsXtrEnableEntryCounting: new ..	80
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new	82
argument	133	\glsxtrscfont: new	188
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new	188
argument	132	\glsxtrsmfont: new	202
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new	202
default type to \acronymtype	93	short-em: new	222
\newterm: fixed name argument	132	short-em-desc: new	224
0.5 (2015-12-07)		short-em-footnote: new	232
\cGLS: new	85	short-em-long: new	219
\cGLS@: new	85	short-em-long-desc: new	220

short-em-postfootnote: new	234	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new	198	attribute	265
short-sc-postfootnote: new	200	\Glsxtrheadtext: now uses headuc	
short-sm: new	205	attribute	266
short-sm-desc: new	207	\glxtrheadtext: now uses headuc	
short-sm-footnote: new	211	attribute	266
short-sm-long: new	204	short-em-footnote: switch off regular	
short-sm-long-desc: new	205	attribute if set	233
short-sm-postfootnote: new	213	short-long: switch off regular attribute	
long-noshort-em: new	225	if set	176
long-noshort-em-desc: new	229	short-long-desc: switch off regular	
long-noshort-sm: new	208	attribute if set	177
long-noshort-sm-desc: new	210	short-sc-footnote: switch off regular	
long-short-em: new	215	attribute if set	198
long-short-em-desc: new	216	short-sm-footnote: switch off regular	
long-short-sm: new	202	attribute if set	212
long-short-sm-desc: new	203	long-short: switch off regular attribute	
0.5.1 (2015-12-02)		if set	173
\Glsaccessstext: new	118	long-short-desc: switch off regular	
0.5.1 (2015-12-07)		attribute if set	175
\@glxtr@doaccsupp: new	16	long-short-sc-desc: switch off regular	
General: removed \ifglxtruseuchead	264	attribute if set	190
\Glsaccessdesc: new	122	footnote: switch off regular attribute if	
\Glsaccessdescplural: new	122	set	178
\Glsaccessfirst: new	119	postfootnote: switch off regular	
\Glsaccessfirstplural: new	120	attribute if set	180
\Glsaccessname: new	118	0.5.2 (2015-12-08)	
\Glsaccessplural: new	119	\@GLSdesc@: added accessibility support	55
\Glsaccesssymbol: new	120	\@GLSdescplural@: added accessibility	
\Glsaccesssymbolplural: new	121	support	55
\glxtrheadfirst: now uses headuc		\@GLSfirst@: added accessibility	
attribute	268	support	52
\glxtrheadfirst: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	268	support	54
\Glsxtrheadfirstplural: now uses		\@GLSname@: added accessibility support	54
headuc attribute	269	\@GLSplural@: added accessibility	
\glxtrheadfirstplural: now uses		support	53
headuc attribute	268	\@GLSsymbol@: added accessibility	
\Glsxtrheadplural: now uses headuc		support	56
attribute	267	\@GLSsymbolplural@: added	
\glxtrheadplural: now uses headuc		accessibility support	56
attribute	267	\@GLStext@: added accessibility support	51
\Glsxtrheadshort: now uses headuc		\@GLSdesc@: added accessibility support	55
attribute	265	\@GLSdescplural@: added accessibility	
\glxtrheadshort: now uses headuc		support	55
attribute	264	\@GLSfirst@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support	52
attribute	266	\@GLSfirstplural@: added accessibility	
		support	54

\@Glsname@: add accessibility support ..	54	\GLSaccesssymbolplural: new ..	121, 126
\@Glsplural@: added accessibility support	53	\GLSaccessstext: new	118, 125
\@Glsymbol@: added accessibility support	56	\glsentryfmt: moved	
\@Glsymbolplural@: added accessibility support	56	\glssetabbrvfmt from	
\@Glstext@: added accessibility support	52	\glsxtrabbrvfmt to here	46
\@glsdesc@: added accessibility support	55	\GlsXtrEnableInitialTagging: new	145
\@glsdescplural@: added accessibility support	55	\glsxtrfieldtitlecase: new	133
\@glsfirst@: added accessibility support	52	\GlsXtrFormatLocationList: new ...	44
\@glsfirstplural@: added accessibility support	53	\glsxtrnewabbrevpresetkeyhook:	
\@Glsname@: added accessibility support	54	new	155
\@Glsplural@: added accessibility support	53	\glsxtrtagfont: new	146
\@Glsymbol@: added accessibility support	56	\KV@printgloss@nonumberlist: added	46
\@Glsymbolplural@: added accessibility support	56	\mfu@checkword@do: added	145
\@Glstext@: added accessibility support	51	\setabbreviationstyle: added check	
\@glsxtr@activate@initialtagging:		for post-definition style switch	170
new	146	0.5.3 (2015-12-09)	
\@glsxtr@do@titlecaps@warn: new ..	146	\@glsxtr@autoindex@at: new	142
\@glsxtr@tag: new	146	\@glsxtr@autoindex@encap: new ...	142
General: fixed typo in glossaries-accsupp		\@glsxtr@autoindex@esc: new	143
and tidied up code to use just one		\@glsxtr@autoindex@level: new ...	143
\@ifpackageloaded	117	\@glsxtr@autoindex@setname: new ..	141
removed \glsxtrabbrvfmt	167	\@glsxtr@doabbreviationsdef: new ..	13
\glossaryentrynumbers: added	43	General: removed	
\Glossentrydesc: added	144	\GlsXtrNoGlsWarningNoAutoMakeMain	
\Glossentryname: added	138	108
\Glossentrysymbol: added	144	\glsdescwidth: added	43
\GLSaccessdesc: new	122, 127	\glspagelistwidth: added	43
\GLSaccessdescplural: new ...	122, 127	\glsxtrdoautoindexname: new	140
\GLSaccessfirst: new	119, 126	\glsxtrpostnamehook: new	139
\GLSaccessfirstplural: new ..	120, 126	\if@glsxtr@format@override: new ..	139
\GLSaccesslong: new	124, 127	\ProvidesGlossariesExtraLang: new	279
\GLSaccesslongpl: new	124, 128	\RequireGlossariesExtraLang: new	279
\Glsaccesslongpl: new	124	0.5.4 (2015-12-15)	
\glsaccesslongpl: new	124	\@newglossaryentry@defunitcounters:	
\GLSaccessname: new	118, 125	new	86
\GLSaccessplural: new	119, 125	\@GLSxtr@p@acrlong@: new	73
\GLSaccesssshort: new	123, 127	\@GLSxtr@p@acrlongpl@: new	74
\GLSaccesssshortpl: new	124, 127	\@GLSxtr@p@acrshort@: new	73
\GLSaccesssymbol: new	121, 126	\@GLSxtr@p@acrshortpl@: new	73
		\@GLSxtr@p@long@: new	72
		\@GLSxtr@p@longpl@: new	73
		\@GLSxtr@p@plural@: new	71
		\@GLSxtr@p@short@: new	72
		\@GLSxtr@p@shortpl@: new	72
		\@GLSxtr@p@text@: new	71
		\@GlsXtrEnableOnTheFly: new	39
		\@Glsxtr: new	40
		\@Glsxtr@p@acrlong@: new	73
		\@Glsxtr@p@acrlongpl@: new	73

\@Glsxtr@p@acrshort@: new	73	\glsenableentryunitcount: new	88
\@Glsxtr@p@acrshortpl@: new	73	\glshasattribute: added check for	
\@Glsxtr@p@long@: new	72	entry's existence	129
\@Glsxtr@p@longpl@: new	73	\glusifattribute: added check for	
\@Glsxtr@p@plural@: new	71	entry's existence	130
\@Glsxtr@p@short@: new	72	\glspostlinkhook: added existence	
\@Glsxtr@p@shortpl@: new	72	check	148
\@Glsxtr@p@text@: new	71	\Glsxtr: new	40
\@Glsxtrpl: new	41	\glxtr: new	39
\@alt@glshyp@opt: new	68	\glxtrcat: new	39
\@glscalt@hyp@opt: new	68	\glxtrdowrglossaryhook: new	68
\@glscalt@hyp@opt@char: new	68	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	68	new	91
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new	39
new	87	\Glsxtrpl: new	41
\@glsc@local@increment@currunitcount:		\glxtrpl: new	40
new	87	\glxtrpostlocalreset: new	80
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new	79
new	65	\glxtrpostreset: new	79
\@glsxtr: new	40	\glxtrpostunset: new	79
\@glsxtr@addunitcounter: new	86	\glxtrprotectlinks: new	70
\@glsxtr@currunitcount: new	88	\GlsXtrSetAltModifier: new	68
\@glsxtr@ifunitcounter: new	86	\GlsXtrSetDefaultGlsOpts: new	67
\@glsxtr@p@acrlong@: new	73	\glxtrstarflywarn: new	39
\@glsxtr@p@acrlongpl@: new	73	\GlsXtrWarning: new	41
\@glsxtr@p@acrshort@: new	73	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new	73	disables \setacronymstyle	93
\@glsxtr@p@long@: new	72	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new	73	\@glsxtr@autoindexcrossrefs: new	12
\@glsxtr@p@plural@: new	71	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new	71	new	100
\@glsxtr@p@shortpl@: new	72	\@glsxtr@idx@entrynumberlist: new	102
\@glsxtr@p@text@: new	71	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new	88	new	100
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new	92	new	101
\@glsxtr@unitcountlist: new	86	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new	40	new	101
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new	95
empty see value if not set and added		\makeglossaries: new	95
'see' to field key map	32	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	39	\glxtrdiscardperiod: added check	
\cGlsformat: added	85	for first use	149
\cGlsformat: added	85	short-desc: fixed typo in	
\cGlsplformat: added	86	\glxtrinlinefullformat and	
\cGlsplformat: added	86	added missing second argument	183
\glstdisablehyper: added	70	1.02 (2016-04-25)	
\glstdohyperlink: added	69	\@glsxtr@current@style: new	42
\glstdonohyperlink: added	70	\Glsfmtfull: new	278

\glsfmtfull: new	277	\@GLSdescplural@: set abbreviation and regular format	55
\Glsfmtfullpl: new	278	\@GLSfirst@: set abbreviation format ..	52
\glsfmtfullpl: new	278	\@GLSfirstplural@: set abbreviation and regular format	54
\Glsfmtlong: new	276	\@GLSname@: set abbreviation and regular format	54
\glsfmtlong: new	276	\@GLSplural@: set abbreviation and regular format	53
\Glsfmtlongpl: new	277	\@GLSsymbol@: set regular format	56
\glsfmtlongpl: new	277	\@GLSsymbolplural@: set regular format	56
\Glsxtrheadfull: new	272	\@GLStext@: set abbreviation and regular format	51
\glsxtrheadfull: new	271	\@GLSuseri@: set regular format	57
\Glsxtrheadfullpl: new	272	\@GLSuserii@: set regular format	57
\glsxtrheadfullpl: new	271	\@GLSuseriii@: set regular format	57
\Glsxtrheadlong: new	270	\@GLSuseriv@: set regular format	58
\glsxtrheadlong: new	269	\@GLSuservi@: set regular format	58
\Glsxtrheadlongpl: new	271	\@GLSdesc@: set abbreviation and regular format	55
\glsxtrheadlongpl: new	270	\@GLSdescplural@: set abbreviation and regular format	55
\Glsxtrtitlefull: new	272	\@GLSfirst@: set abbreviation and regular format	52
\glsxtrtitlefull: new	271	\@GLSfirstplural@: set abbreviation and regular format	54
\Glsxtrtitlefullpl: new	273	\@GLSname@: set abbreviation and regular format	54
\glsxtrtitlefullpl: new	272	\@GLSplural@: set abbreviation and regular format	53
\Glsxtrtitlelong: new	270	\@GLSsymbol@: set regular format	56
\glsxtrtitlelong: new	270	\@GLSsymbolplural@: set regular format	56
\Glsxtrtitlelongpl: new	271	\@GLStext@: set abbreviation and regular format	52
\glsxtrtitlelongpl: new	270	\@GLSuseri@: set regular format	57
\ifglsxtrinsertinside: new	173	\@GLSuserii@: set regular format	57
postfootnote: added redef of		\@GLSuseriii@: set regular format	57
\glsxtrsetupfulldefs	180	\@GLSuseriv@: set regular format	57
stylemods: new	17	\@GLSuservi@: set regular format	58
1.03 (2016-04-27)		\@GLSdesc@: set abbreviation and regular format	55
\@GLSfirstplural@: bug fix: misspelt cs name	54	\@GLSdescplural@: set abbreviation and regular format	53
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	53	\@GLSsymbol@: set regular format	56
\@GLSfirstplural@: bug fix: misspelt cs name	54	\@GLSsymbolplural@: set regular format	56
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	53	\@GLStext@: set abbreviation and regular format	52
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	53	\@GLSuseri@: set regular format	57
\glsxtrtitlelongpl: bug fix: changed \glsxtrlong to \glsxtrlongpl ..	270	\@GLSuserii@: set regular format	57
\glsxtrtitleshortpl: bug fix: changed \glsxtrshort to \glsxtrshortpl	265	\@GLSuseriii@: set regular format	57
1.04 (2015-04-30)		\@GLSuseriv@: set regular format	57
short-em-footnote: renamed from “footnote-em”	232	\@GLSuservi@: set regular format	58
1.04 (2016-05-02)		\@GLSdesc@: set abbreviation and regular format	55
\@glsxtrpostloctag: new	45	\@GLSdescplural@: set abbreviation and regular format	55
\@GLSdesc@: set abbreviation and regular format	55	\@GLSfirst@: set abbreviation and regular format	52

\@glsfirstplural@: set abbreviation and regular format	53	\glxtruserparen: new	236
\@glsname@: set abbreviation and regular format	54	\glxtrusersuffix: new	237
\@glsplural@: set abbreviation and regular format	53	\GlsXtrWarnDeprecatedAbbrStyle: new	172
\@glssymbol@: set regular format	56	short-em-long-em: new	220
\@glssymbolplural@: set regular format	56	short-em-long-em-desc: new	222
\@glstext@: set abbreviation and regular format	51	short-em-nolong: new	224
\@glxtr@deprecated@abbrstyle: new	172	short-em-nolong-desc: new	225
\@glxtr@do@style: new	18	short-em-postfootnote: renamed from “postfootnote-em”	234
\@glxtr@doloctag: new	45	short-footnote: new	179
\@glxtr@idx@entrynumberlist: switched from \let to \newcommand	102	short-long-user: new	243
\@glxtr@pagetag: new	45	short-long-user-desc: new	244
\@glxtr@pagetag: new	45	short-nolong: new	183
\@glxtr@preloctag: new	45	short-nolong-desc: new	184
\@glxtr@postloctag: new	45	short-postfootnote: new	181
\@glxtr@preloctag: new	44, 45	short-sc-footnote: renamed from “footnote-sc”	198
\glossentrydesc: added glossdescfont attribute check	134	short-sc-nolong: new	193
\Glossentryname: added glossnamefont attribute check	138	short-sc-nolong-desc: new	195
\glossentryname: added glossnamefont attribute check	136	short-sc-postfootnote: renamed from “postfootnote-sc”	200
moved post name hook inside condition	138	short-sm-footnote: renamed from “footnote-sm”	211
\glsabbrvmfont: new	215	short-sm-nolong: new	207
\glsabbrvuserfont: new	236	short-sm-nolong-desc: new	208
\glsfirstabbrvmfont: new	215	short-sm-postfootnote: renamed from “postfootnote-sm”	213
\glsfirstabbrvuserfont: new	236	\letabbreviationstyle: new	172
\glsfirstlongemfont: new	215	\newabbreviationstyle: bug fix: corrected test for existence	171
\glsfirstlonguserfont: new	237	long-em-noshort-em: new	227
\glsifnotregularcategory: new ...	130	long-em-noshort-em-desc: new ...	230
\glslongdefaultfont: new	157	long-em-short-em: new	217
\glslongemfont: new	215	long-em-short-em-desc: new	218
\glslongfont: new	157	long-noshort: new	187
\glslonguserfont: new	236	long-noshort-desc: new	187
\glxtrassignfieldfont: new	51	long-noshort-em: renamed from “long-em”	225
\GlsXtrEnablePreLocationTag: new .	44	long-noshort-em-desc: renamed from “long-desc-em”	229
\glxtrfirstscfont: new	188	long-noshort-sc: renamed from “long-sc”	195
\glxtrfirstsmfont: new	202	long-noshort-sc-desc: renamed from “long-desc-sc”	196
\glxtrlongshortdescsort: new ...	174	long-noshort-sm: renamed from “long-sm”	208
\glxtrpostnamehook: added category check	139	long-noshort-sm-desc: renamed from \long-desc-sm	210
\glxtrregularfont: new	46		
\glxtruserfield: new	236		

long-short-user: new	237	docdef option changed to choice	11
long-short-user-desc: new	242	\glxtr@usesee: new	32
\renewabbreviationstyle: new	171	\glxtrusesee: new	32
style: new	18	\glxtruseseeformat: new	32
1.05 (2016-06-10)		\if@glxtrdocdefrestricted: new ..	12
\eglssetwidest: new	287	1.07 (2016-08-15)	
\glsFindWidestAnyName: new	289	\@@glxtrp: new	74
\glsFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new	294	nohyperfirst attribute	53
\glsFindWidestAnyNameSymbol: new	292	\@GLSfirstplural@: added check for	
\glsFindWidestAnyNameSymbolLocation:		nohyperfirst attribute	54
new	293	\@GLSxtrp: new	75
\glsFindWidestLevelTwo: new	290	\@Glsfirst@: added check for	
\glsFindWidestUsedAnyName: new ..	288	nohyperfirst attribute	52
\glsFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new	294	nohyperfirst attribute	54
\glsFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new	74
new	291	\@gls@preglossaryhook: added	
\glsFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts	147
new	292	\@glsfirst@: added check for	
\glsFindWidestUsedLevelTwo: new .	289	nohyperfirst attribute	52
\glsFindWidestUsedTopLevelName:		\@glsfirstplural@: added check for	
new	288	nohyperfirst attribute	53
\glsfirstlongfootnotefont: new ..	177	\@glxtrinmark: new	263
\glsgetwidestname: new	287	\@glxtrnotinmark: new	263
\glsgetwidestsubname: new	288	\@glxtrp: new	74
\glslongfootnotefont: new	177	\@glxtrp@opt: new	74
\glxtrAltTreeIndent: new	287	\glossxtrsetpopts: new	74
\glxtralttreeInit: new	287	\glsp: new	76
\glxtrAltTreePar: new	287	\glsp: new	76
\glxtrAltTreeSetHangIndent: new	295	\glxtr@entry@p: new	75
\glxtrAltTreeSetSubHangIndent:		\glxtrabbrvfootnote: new	177
new	295	\glxtrchecknohyperfirst: new	52
\glxtralttreeSubSymbolDescLocation:		\glxtrfieldtitlecasecs: new	134
new	287	\glxtrifinmark: new	262
\glxtralttreeSymbolDescLocation:		\GLSxtrp: new	78
new	286	\Glsxtrp: new	77
\glxtrComputeTreeIndent: new ...	295	\glxtrp: new	75
\glxtrComputeTreeSubIndent: new	295	\glxtrsetpopts: new	74
\glxtrtreetopindent: new	287	short-long-desc: added text key	177
short-em-long: fixed incorrect font used		fixed misspelling of \glsabbrvfont in	
by long form	219	plural key	177
\xglsetwidest: new	287	long-short-desc: added missing text	
1.06 (2016-06-18)		key	175
\@glsdoifexistsorwarn: new	12	fixed misspelling of \glsabbrvfont .	175
\@glxtr@docdefval: new	11	footnote: changed first forms to use	
\@glxtr@usesee: new	32	\glsfirstlongfootnotefont ...	178
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document	20	from first keys	179

switched from \glsfirstlongfont to	1.10 (2016-12-17)
\glsfirstlongfootnotefont ... 181	\@GLSpl@: fixed bug caused by typo in
\RestoreAcronyms: modified	command name 48
\@gls@link@checkfirsthyper to	1.11 (2017-01-19)
set \glsxtrifwasfirstuse 94	\@glsxtr@do@redef@for@gl@sentries:
1.08 (2016-12-13)	new 6
\@glsxtr@record: new 7	\@glsxtr@noidx@do: new 116
\@GLS@: added \@glsxtr@record 48	\@glsxtr@redef@for@gl@sentries: new . 6
\@GLSpl@: added \@glsxtr@record ... 48	\@glsxtr@shortcutsval: new 15
\@Gls@: added \@glsxtr@record 47	\@glsxtr@unsrt@getgroup@title: new 116
\@GLspl@: added \@glsxtr@record ... 48	\@print@noidx@glossary: added
\@Gls@: added \@glsxtr@record 47	redefinition 103
\@Gls@@link@: added	\glsxtr@addloc@listfield: added
\@glsxtr@record 48	group key 10
\@Gls@field@link: added	added location key 10
\@glsxtr@record 47	\glsxtr@fields: new 111
\@Gls@saveentrycounter: new 20	\glsxtr@linkprefix: new 111
\@Glsdisp: added \@glsxtr@record .. 48	\glsxtr@org@new@ignored@glossary:
\@GLspl@: added \@glsxtr@record ... 47	new 27
\@glsxtr@dorecord: new 8	\glsxtr@s@new@ignored@glossary: new 28
\@glsxtr@err@undefaction: new 6	\glsxtr@shortcutsval: new 111
\@glsxtr@record: new 7	\glsxtr@texencoding: new 111
\@glsxtr@warn@onexists@do: new ... 6	\glsxtr@writefields: new 112
\@glsxtr@warn@undefaction: new 6	\GlsXtrLoadResources: new 111
\@print@unsrt@glossary: new 114	\glsxtr@resourcefile: changed
General: added record package option ... 10	extension to .glstex 110
\glsadd: added \@glsxtr@record 50	\new@ignored@glossary: added starred
\glsdoifexists: now defines	version 27
\glslabel 30	1.12 (2017-02-03)
\glsxtr@do@wrglossary: new 19	\@glsxtr@recordcounter: new 8
\glsxtr@addloc@listfield: new 9	\@Gls@preglossaryhook: check for
\glsxtr@indexonly@saveentrycounter:	definition 147
new 9	\@glsxtr@counter@recordhook: new . 113
\glsxtr@record: new 113	\@glsxtr@display@loc: new 104
\glsxtr@resource: new 111	\@glsxtr@docounter@record: new ... 113
\glsxtr@saveentrycounter: new 20	\@glsxtr@longnew@glossary@entry:
\glsxtr@setup@record: new 9	new 26
\glsxtr@assignfieldfont: added check	\@glsxtr@noop@recordcounter: new .. 8
for existence 51	\@glsxtr@op@recordcounter: new 9
\glsxtr@resourcefile: new 110	\@glsxtr@provide@storagekey: new . 21
\printunsrt@glossaries: new 114	\@glsxtr@s@longnew@glossary@entry:
\printunsrt@glossary: new 113	new 26
1.09 (2016-12-16)	\@glsxtr@entryfmt: new 22
\@glsxtr@gettype: new 100	\@glsxtr@index@aliased: new 66
\@glsxtr@mixed@assign@sortkey:	\@glsxtr@set@alias@noindex: new 66
new 100	\@new@glossary@entry@posthook: added
\@print@glossary: redefined to save	check for alias key 36
options 99	\@no@glsxtr@index@aliased: new 66
\glsxtr@makeglossaries: new 100	\@print@unsrt@glossary: new 114

General: added target key to printgloss family	99	\GlsXtrLoadResources: removed restriction on only one per document	111
\apptoglossarypreamble: new	25	\glxtrlocrangefmt: new	105
\csGlsXtrLetField: new	24	\glxtrpostlongdescription: new ..	27
\eGlsXtrSetField: new	25	\glxtrprovidestoragekey: new	20
\gGlsXtrSetField: new	25	\GlsXtrRecordCounter: new	113
\glsdohyperlink: added check for alias field	69	\glxtrresourcecount: new	111
\glsnoidxdisplayloc: added redefinition	104	\glxtrresourcefile: added catcode change for @	111
\glissettoctitle: added patch	28	\glxtrsetaliasnoindex: new	66
\glxtr@counterrecord: new	113	\GlsXtrSetField: new	24
\glxtr@langtag: new	111	\glxtrsetfieldifexists: new	24
\glxtr@newabbreviation: new	153	\glxtrunsrtdo: new	116
\glxtr@org@newignoredglossary: Added check for existence	27	\Glsxtrusefield: new	24
\glxtr@pluralsuffixes: new	111	\glxtrusefield: new	24
\glxtr@provideignoredglossary: new	29	short-postlong-user: new	240
\glxtr@s@newignoredglossary: Added check for existence	28	short-postlong-user-desc: new ...	242
\glxtr@s@provideignoredglossary: new	29	\longnewglossaryentry: added starred version	26
\glxtrabbrvpluralsuffix: new ...	157	long-postshort-user: new	238
\glxtralias: new	35	long-postshort-user-desc: new ...	239
\glxtrcopytoglossary: new	30	postdot: new	13
\glxtrdeffield: new	24	\pretoglossarypreamble: new	25
\glxtrdisplayendloc: new	105	\print@noop@unsrtglossaryunit: new	115
\glxtrdisplayendlochook: new ...	105	\print@op@unsrtglossaryunit: new	115
\glxtrdisplaysingleloc: new	104	\printunsrtglossary: added starred form	113
\glxtrdisplaystartloc: new	104	\printunsrtglossaryhandler: new .	115
\glxtrdeffield: new	24	\printunsrtglossaryunit: new	9
\glxtrentryfmt: new	22	\printunsrtglossaryunitsetup: new	115
\glxtrfielddolistloop: new	23	\provideignoredglossary: new	29
\glxtrfieldforlistloop: new	23	\s@glxtr@provide@storagekey: new	21
\glxtrfieldifinlist: new	23	\s@printunsrtglossary: new	114
\glxtrfieldlistadd: new	23	\xGlsXtrSetField: new	25
\glxtrfieldlistead: new	23	1.13 (2017-02-07)	
\glxtrfieldlistgadd: new	23	\@glsdisp: removed	
\glxtrfieldlistxadd: new	23	\@glxtr@org@glsdisp	48
\glxtrfieldxifinlist: new	23	\glxtrsetaliasnoindex: switched to \providecommand	66
\glxtrfmt: new	22	1.14 (2017-04-18)	
\GlsXtrFmtDefaultOptions: new	22	\@gls@link: added redefinition	49
\GlsXtrFmtField: new	22	\@gls@noidx@getgrouptitle: new ..	102
\glxtrifkeydefined: new	20	\@gls@removespaces: new	105
\glxtrindexaliased: new	67	\@glxtr@do@automake@err: new ...	113
\GlsXtrLetField: new	24	\@glxtr@org@gloautosee: new	18
\GlsXtrLetFieldToField: new	24	\@glxtr@record: added third arg	7
		\@glxtr@recordsee: new	9

General: added \glsadd option		1.16 (2017-06-15)	
theHvalue	50	\@glo@autosee: added redefinition	19
added \glsadd option thevalue	50	\@gls@noidx@getgrouptitle: fixed	
\glsdisablehyper: added redefinition .	69	bug	102
\glsenableentrycount: fixed		\@glsxtr@addunusedxrefs: added	
assignment of \@cGls@	81	check for seealso field	37
\glsenableentryunitcount: fixed		\@glsxtr@dorecordnodefer: new	8
assignment of \@cGls@	90	\@glsxtr@noidx@do: use \csuse instead	
\glsnavigation: new	103	of \csname	116
\glsxtr@org@getgrouptitle: new ..	102	\@print@unsrt@glossary: corrected	
\glsxtr@recordsee: new	7	misspelt command	114
\glsxtr@writefields: added check for		\@printunsrt@glossary@handler:	
automake	112	new	115
\glsxtrdisplayendloc: added check		General: added check for	
for empty format	105	\@gls@setupsort@none	11
\glsxtrgetgrouptitle: new	102	\@gls@checkseeallowed: added	
\glsxtrinitwrgloss: new	49	redefinition	19
\glsxtrlocationhyperlink: new ...	105	\@glsxtr@writefields: added	
\glsxtrsetgrouptitle: new	103	\providecommand lines	112
\glsxtrsupphypernumber: new	106	\glsxtrautoindex: new	141
\ifglsxtrwrglossbefore: new	49	\glsxtrautoindexentry: new	141
1.15 (2017-05-10)		\glsxtrautoindexsort: new	141
\@glsxtr@dorecord: corrected		\glsxtrindexseealso: new	33
premature expansion of \@glslocref	8	\glsxtrseealsolabels: new	36
short-em-long-em: fixed spelling of		\glsxtrseelist: new	33
\glsabbrvfont	221	\glsxtruseseealso: new	33
short-long: fixed spelling of		\glsxtruseseealsoformat: new	33
\glsabbrvfont	175	\seealso name: new	33
short-long-user: fixed spelling of		autoseeindex: new	12
\glsabbrvfont	243	1.17 (2017-08-09)	
short-postlong-user: fixed spelling of		\@glsxtr@mark@wordseps: new	153
\glsabbrvfont	240	\@glsxtr@markwordseps: new	152
short-postlong-user-desc: fixed		\@glsxtr@noidx@displaynumberlist:	
spelling of \glsabbrvfont	242	replace hard-coded ?? with	
long-em-short-em: fixed spelling of		\glsxtrundeftag	101
\glsabbrvfont	217	\@glsxtr@noidx@entrynumberlist:	
long-postshort-user: fixed spelling of		replace hard-coded ?? with	
\glsabbrvfont	238	\glsxtrundeftag	101
long-postshort-user-desc: fixed		\@glsxtr@noidx@numberlistloop:	
spelling of \glsabbrvfont	240	replace hard-coded ?? with	
long-short: fixed spelling of		\glsxtrundeftag	101
\glsabbrvfont	173	\@glsxtrifhyphenstart: new	245
long-short-user: fixed spelling of		General: removed some inconsistencies	
\glsabbrvfont	237	in the abbreviation styles	173
footnote: fixed spelling of		\glsabbrvhyphenfont: new	246
\glsabbrvfont	178	\glsabbrvonlyfont: new	259
postfootnote: fixed spelling of		\glsabbrvscfont: new	188
\glsabbrvfont	180	\glsabbrvsmfont: new	202

\glsabbrvuserfont: initialised to default font	236	\glsxtrshortlongdescsort: new ...	176
\glsfirstabbrvhyphenfont: new ...	246	\Glsxtrsubsequentfmt: new	169
\glsfirstabbrvonlyfont: new	259	\glsxtrsubsequentfmt: new	169
\glsfirstabbrvscfont: new	188	\Glsxtrsubsequentplfmt: new	170
\glsfirstabbrvsmfont: new	202	\glsxtrsubsequentplfmt: new	169
\glsfirstlonghyphenfont: new ...	246	\glsxtrword: new	152
\glsfirstlongonlyfont: new	259	\glsxtrwordsep: new	152
\glslonghyphenfont: new	246	short-hyphen-long-hyphen: new ...	254
\glslongonlyfont: new	259	short-hyphen-long-hyphen-desc: new	255
\glslonguserfont: initialised to default font	236	short-hyphen-postlong-hyphen: new	256
\glsxtr@newabbreviation: added \glsxtrorgshort and \glsxtrorglong	153	short-hyphen-postlong-hyphen-desc: new	258
\GlsXtrDefineAcShortcuts: new	14	short-long-user-desc: corrected first forms	244
\glsxtrgenabbrvfmt: added check for \ifglsxtrininsertinside	167	short-nolong-desc-noreg: new	185
\glsxtrhyphensuffix: new	246	short-nolong-noreg: new	183
\glsxtrifhyphenstart: new	245	long-em-noshort-em-desc-noreg: new	232
\glsxtrlonghyphen: new	250	long-em-noshort-em-noreg: new ...	228
\glsxtrlonghyphennoshort: new ...	248	long-hyphen-noshort-desc-noreg: new	248
\glsxtrlonghyphenshort: new	245	long-hyphen-postshort-hyphen: new	251
\glsxtrlongshortdescname: new ...	174	long-hyphen-postshort-hyphen-desc: new	253
\glsxtronlydescname: new	261	long-hyphen-short-hyphen: new ...	246
\glsxtronlydescsort: new	261	long-hyphen-short-hyphen-desc: new	247
\glsxtronlysuffix: new	259	long-noshort-desc-noreg: new	187
\glsxtrparen: new	155	long-noshort-noreg: new	188
\glsxtrposthyphenlong: new	256	long-only-short-only: new	259
\glsxtrposthyphenshort: new	250	long-only-short-only-desc: new ..	261
\glsxtrposthyphensubsequent: new	251	long-short-user-desc: corrected first forms	243
\glsxtrshortdescname: new	183		
\glsxtrshorthyphen: new	256		
\glsxtrshorthyphenlong: new	254		
\glsxtrshortlongdescname: new ...	176		

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	
\.	148, 286
\@	111
\@cGLS@	81, 90
\@cGLSpl@	81, 90
\@cGls@	81, 90
\@cGlspl@	81, 90
\@cgls@	81, 90
\@cglspl@	81, 83, 90
\@do@wrglossary	8, 96
\@do@wrglossary	10, 11, 19, 51, 66
\@glo@assign@sortkey	100
\@glo@list	6
\@glo@type	114
\@gls@expand@field	21
\@glslocalreset	79
\@glslocalunset	79
\@glsreset	79
\@glsunset	79
\@glsxtr@autoindex@escspch	142–144
\@glsxtr@checkspch	141, 142, 144
\@glsxtr@disabledflycommand	42
\@glsxtr@record	11
\@glsxtr@recordcounter	10, 11, 113
\@glsxtrp	74, 75
\@glsxtrpostloctag	44
\@glsxtrpreloctag	44, 45
\@newglossaryentry@defcounters	80
\@newglossaryentry@defunitcounters	88
\@par	287
\@ACRlong	71
\@ACRlongpl	71
\@ACRshort	71
\@ACRshortpl	71
\@Acrlong	71
\@Acrlongpl	71
\@Acrshort	71
\@Acrshortpl	71
\@GLS@	70, 84, 85
\@GLSdesc@	55
\@GLSpl@	70, 84, 85
\@GLSplural@	71
\@GLSsymbol@	56
\@GLStext@	71
\@GLSxtr@full	159
\@GLSxtr@fullpl	160
\@GLSxtr@p@acrlong@	71
\@GLSxtr@p@acrlongpl@	71
\@GLSxtr@p@acrshort@	71
\@GLSxtr@p@acrshortpl@	71
\@GLSxtr@p@long@	70
\@GLSxtr@p@longpl@	71
\@GLSxtr@p@plural@	70
\@GLSxtr@p@short@	70
\@GLSxtr@p@shortpl@	71
\@GLSxtr@p@text@	70
\@GLSxtrlong	70, 163
\@GLSxtrlongpl	71, 166, 167
\@GLSxtrp	78, 79
\@GLSxtrshort	70, 162
\@GLSxtrshortpl	71, 165
\@Gls@	70, 83, 84
\@Gls@acentryname	92
\@Gls@entry@field	63, 77
\@Gls@entryname	92
\@GlsXtrEnableOnTheFly	39
\@Glspl@	70, 84
\@Glsplural@	71
\@Glstext@	71
\@Glsxtr	40, 41
\@Glsxtr@full	158
\@Glsxtr@fullpl	160
\@Glsxtr@p@acrlong@	71
\@Glsxtr@p@acrlongpl@	71

\@Glsxtr@p@acrshort@	71	\@glo@assign@sortkey	98
\@Glsxtr@p@acrshortpl@	71	\@glo@autosee	18
\@Glsxtr@p@long@	70	\@glo@autoseehook	35
\@Glsxtr@p@longpl@	71	\@glo@category	86
\@Glsxtr@p@plural@	70	\@glo@check@sortallowed	98
\@Glsxtr@p@short@	70	\@glo@counterprefix	8, 105
\@Glsxtr@p@shortpl@	71	\@glo@countunit	86
\@Glsxtr@p@text@	70	\@glo@default@sorttype	98
\@Glsxtrlong	70, 163	\@glo@desc	26, 27
\@Glsxtrlongpl	71, 166	\@glo@descplural	26, 27
\@Glsxtrp	77	\@glo@group	10
\@Glsxtrpl	41	\@glo@label	9, 10, 21, 32, 34–36, 63, 70, 288–295
\@Glsxtrshort	70, 161	\@glo@location	10
\@Glsxtrshortpl	71, 164	\@glo@loclist	9
\@acrlong	71	\@glo@name	140, 141
\@acrlongpl	71	\@glo@no@assign@sortkey	100
\@acrshort	71	\@glo@parent	289–291
\@acrshortpl	71	\@glo@see	32, 33, 35–37
\@alt@glshyp@opt	68	\@glo@seealso	34, 35
\@auxout	8, 9, 45, 82, 90, 91, 95, 96, 106, 107, 110, 112, 113	\@glo@sort	141
\@bibgls@restreat	111	\@glo@sorttype	98, 103
\@cGLS	85	\@glo@text	48
\@cGLS@	81, 85, 90	\@glo@thislettergrp	116, 117
\@cGLSpl	85	\@glo@thisvalue	236
\@cGLSpl@	81, 85, 90	\@glo@tmp	21, 33, 63
\@cGls@	81, 90	\@glo@type	36, 92, 95, 98–100, 103, 104, 106, 107, 109, 110, 114
\@cGlspl@	81, 90	\@glo@types	131, 132, 288–294
\@cgls@	81, 90	\@glossary@default@style	42, 98, 297
\@cglspl@	81, 90	\@glossarystyle	98, 99
\@disable@onlypremakeg	96	\@gls@	70, 83, 84
\@do@auxoutstuff	106, 107	\@gls@@link	48
\@do@glsee	35, 36	\@gls@ReturnAfterFi	105
\@do@newglossaryentry	92, 93, 155	\@gls@actualchar	141
\@do@seeglossary	10, 11, 18, 96	\@gls@adjustmode	50
\@do@wrglossary	50	\@gls@alt@hyp@opt	68
\@empty	51, 59–62, 141, 142, 158–167	\@gls@alt@hyp@opt@char	68
\@end@glxtr@addunused	37	\@gls@alt@hyp@opt@keys	68
\@end@glxtr@gettype	98, 100	\@gls@automake	98
\@end@glxtr@usesee	32	\@gls@between	103
\@end@glxtr@trifhyphenstart	245	\@gls@checkedmkidx	141, 142, 144
\@endfortrue	170	\@gls@checkmkidxchars	34, 141
\@firstofone	51, 134, 135, 140, 146	\@gls@codepage	107
\@firstofthree	48, 51, 59–62, 68, 158, 159, 161, 162, 164, 166	\@gls@counter	7, 8, 49, 50, 66
\@firstoftwo	52–56, 60, 62, 65, 68, 94, 149, 150, 158–160, 164–167, 263	\@gls@currentlettergroup	103, 114, 116, 117
\@for	6, 17, 37, 80, 92, 95, 98, 103, 114, 133, 145	\@gls@declareoption	5
\@glo@alias	34, 35	\@gls@default@longpl	153, 154
		\@gls@doautomake	98, 113
		\@gls@doautomake@err	113

\@gls@encapchar	142	\@gls@shortpl	151, 154, 155
\@gls@entry@count	82	\@gls@sort	116
\@gls@entry@field .. 21, 24, 35, 63, 75–78, 81		\@gls@tmp	103
\@gls@entry@unitcount	90, 91	\@gls@tmpb	144
\@gls@field@font	51–58	\@gls@type	96, 98, 170, 288–294
\@gls@field@link	51–58, 63, 64	\@gls@write@entrycounts	82
\@gls@getgrouptitle	102, 103, 114	\@gls@write@entryunitcounts	90
\@gls@grptitle	103	\@gls@write@entryunitcounts@do	91
\@gls@hyp@opt	63, 64, 68, 85, 157–166	\@gls@xref	9, 34
\@gls@hyp@opt@cs	68	\@glsabbrv@current@abbreviation	153, 167
\@gls@increment@currcount	81	\@glsacronymlists	92
\@gls@increment@currunitcount	89	\@glsdoifexistsorwarn	12, 136–139
\@gls@keymap	9, 10, 21, 32, 34, 63, 112	\@glsentry	82, 90, 91
\@gls@label	8, 67, 68, 96, 113, 170	\@glslink	50, 70
\@gls@levelchar	141	\@glslocref	8
\@gls@link	22, 47–49, 59–63, 158–167	\@glsnextpages	99
\@gls@link@checkfirsthyper	48, 94	\@glsnonextpages	99
\@gls@link@label	49	\@glsnumberformat .. 7, 8, 49, 50, 66, 139, 140	
\@gls@link@nocheckfirsthyper		\@glsorder	95
..... 47, 59–62, 158–167		\@glspl@	70, 83, 84
\@gls@link@opts	49	\@glsplural@	71
\@gls@list	103	\@glsplunc@token	150
\@gls@local@increment@currcount 81		\@glsstyle@alttree	286
\@gls@local@increment@currunitcount	89	\@glsstyle@inline	286
\@gls@location	116, 117	\@glsstyle@listdotted	281
\@gls@loclist	100, 101, 116, 117	\@glsstyle@listdotted	281
\@gls@long	153, 154	\@glsstyle@listdotted	281
\@gls@longpl	151, 153–155	\@glsstyle@listdotted	281
\@gls@nohyperlist	27–29	\@glsstyle@listdotted	281
\@gls@noidx@do	103	\@glsstyle@listdotted	281
\@gls@noidx@getgrouptitle	114	\@glsstyle@listdotted	281
\@gls@noidx@nosanitizesort	97	\@glsstyle@listdotted	281
\@gls@noidx@sanitizesort	97	\@glsstyle@listdotted	281
\@gls@noidx@loclist@finalsep	100	\@glsstyle@listdotted	281
\@gls@noidx@loclist@prev	100	\@glsstyle@listdotted	281
\@gls@noidx@loclist@sep	100	\@glsstyle@listdotted	281
\@gls@noref@warn	96, 104	\@glsstyle@listdotted	281
\@gls@org@glsnoidx@displayloc	101	\@glsstyle@listdotted	281
\@gls@org@glsseeformat	101	\@glsstyle@listdotted	281
\@gls@preglossaryhook	99, 145	\@glsstyle@listdotted	281
\@gls@prevlevel	296	\@glsstyle@listdotted	281
\@gls@quotechar	141	\@glsstyle@listdotted	281
\@gls@reference	38, 95, 96	\@glsstyle@listdotted	281
\@gls@saveentrycounter	10, 11, 20, 50	\@glsstyle@listdotted	281
\@gls@see@noindex	19, 110, 111	\@glsstyle@listdotted	281
\@gls@setdefault@glslink@opts ... 50, 67		\@glsstyle@listdotted	281
\@gls@setsort	50	\@glsstyle@listdotted	281
\@gls@setupsort@none	11	\@glsstyle@listdotted	281
\@gls@short	154	\@glsstyle@listdotted	281

\@glsxtr@autoseeindextrue	12	\@glsxtr@glossdescfont	134, 135
\@glsxtr@cat	80, 92, 145	\@glsxtr@glossnamefont	136–139
\@glsxtr@counterrecordhook	8, 9	\@glsxtr@gobbleto@endescspch	144
\@glsxtr@csname	87–89	\@glsxtr@idx@displaynumberlist	97
\@glsxtr@current@style	42, 297	\@glsxtr@idx@entrynumberlist	97
\@glsxtr@currentunitcount	87, 89	\@glsxtr@ifcsstart	39
\@glsxtr@currunitcount	88, 90	\@glsxtr@ifpunctoken	150
\@glsxtr@declareoption	5, 13, 17	\@glsxtr@ifunitcounter	86
\@glsxtr@defaultnoglossarywarning ..	17	\@glsxtr@insert@dots	152
\@glsxtr@deprecated@abbrstyle	196, 198, 200,	\@glsxtr@insert@dots@next	152
201, 210, 211, 213, 215, 227, 230, 234, 236		\@glsxtr@insertdots	154
\@glsxtr@disabledflycommand	41	\@glsxtr@label	37, 133
\@glsxtr@display@loc	104	\@glsxtr@loadstyles	280
\@glsxtr@do@wrindex	67	\@glsxtr@longnewglossaryentry	26
\@glsxtr@do@gl:disablehyperinlist ..	65	\@glsxtr@mark@wordseps	152
\@glsxtr@do@redef@for:gl:sentries	7	\@glsxtr@mark@wordseps@next	153
\@glsxtr@do@style	18, 279	\@glsxtr@markwordseps	153–155
\@glsxtr@do@titlecaps@warn ..	134–137, 146	\@glsxtr@mixed@assign@sortkey	98
\@glsxtr@do@abbreviationsdef	13	\@glsxtr@noidx@displaynumberlist ...	97
\@glsxtr@do@accsupp	17, 18	\@glsxtr@noidx@do	116
\@glsxtr@docdefval	11, 12, 38	\@glsxtr@noidx@entrynumberlist	97
\@glsxtr@docounterrecord	9	\@glsxtr@noidx@numberlistloop	97
\@glsxtr@doglossary	114, 115	\@glsxtr@noop@recordcounter	8, 10
\@glsxtr@doloctag	44, 45	\@glsxtr@notfoundinlist	150
\@glsxtr@dorecord	8	\@glsxtr@op@recordcounter	11
\@glsxtr@dorecordnodefer	8	\@glsxtr@optlist	41
\@glsxtr@dostylewarn	170	\@glsxtr@org@GLS@	48
\@glsxtr@enabletagging	145	\@glsxtr@org@GLSpl@	48
\@glsxtr@end@	39	\@glsxtr@org@Gls@	47
\@glsxtr@endescspch	141–144	\@glsxtr@org@Glspl@	48
\@glsxtr@entrycount@org@localreset ..	81	\@glsxtr@org@Glsxtr:titlefirst ..	263, 264
\@glsxtr@entrycount@org@localunset ..	81	\@glsxtr@org@Glsxtr:titlefirstplural	263, 264
\@glsxtr@entrycount@org@reset	81	263, 264
\@glsxtr@entrycount@org@unset	81	\@glsxtr@org@Glsxtr:titlefull ..	263, 264
\@glsxtr@entryunitcount@org@localreset	89	\@glsxtr@org@Glsxtr:titlefullpl	263, 264
.....	89	\@glsxtr@org@Glsxtr:titlelong ..	263, 264
\@glsxtr@entryunitcount@org@reset ..	89	\@glsxtr@org@Glsxtr:titlelongpl	263, 264
\@glsxtr@entryunitcount@org@unset ..	89	\@glsxtr@org@Glsxtr:titleplural	263, 264
\@glsxtr@err@undefaction	7, 10	\@glsxtr@org@Glsxtr:titleshort ..	263, 264
\@glsxtr@field@linkdefs	47	\@glsxtr@org@Glsxtr:titleshortpl	263, 264
\@glsxtr@format@overridefalse	139	\@glsxtr@org@Glsxtr:titletext ..	263, 264
\@glsxtr@format@override: true	140	\@glsxtr@org@MakeUppercase	263, 264
\@glsxtr@foundinlist	150	\@glsxtr@org@checkfirsthyper	65, 94
\@glsxtr@full	157	\@glsxtr@org@delimN	45
\@glsxtr@fullpl	159	\@glsxtr@org@delimR	45
\@glsxtr@gettype	98	\@glsxtr@org@doseeglossary	10, 11, 96
		\@glsxtr@org@gloautosee	18, 19
		\@glsxtr@org@gls@	47
		\@glsxtr@org@glsignore	45

<code>\@glxtr@org@glspl@</code>	47	<code>\@glxtr@swaptwo</code>	151
<code>\@glxtr@org@glxtrtitlefirst</code> ..	263, 264	<code>\@glxtr@tag</code>	145
<code>\@glxtr@org@glxtrtitlefirstplural</code>		<code>\@glxtr@taggingcs</code>	145
.....	263, 264	<code>\@glxtr@theHvalue</code>	7, 8, 50, 51
<code>\@glxtr@org@glxtrtitlefull</code> ..	263, 264	<code>\@glxtr@thevalue</code>	7, 8, 50, 51
<code>\@glxtr@org@glxtrtitlefullpl</code>	263, 264	<code>\@glxtr@thisloctag</code>	45
<code>\@glxtr@org@glxtrtitlelong</code> ..	263, 264	<code>\@glxtr@titlelabel</code>	102, 103, 116
<code>\@glxtr@org@glxtrtitlelongpl</code>	263, 264	<code>\@glxtr@tmp</code>	17, 105
<code>\@glxtr@org@glxtrtitleplural</code>	263, 264	<code>\@glxtr@type</code>	133
<code>\@glxtr@org@glxtrtitleshort</code> ..	263, 264	<code>\@glxtr@unitcountlist</code>	86
<code>\@glxtr@org@glxtrtitleshortpl</code>	263, 264	<code>\@glxtr@unsrt@getgrouptitle</code>	114
<code>\@glxtr@org@glxtrtitletext</code> ..	263, 264	<code>\@glxtr@usesee</code>	32
<code>\@glxtr@org@makeglossaries</code>	95	<code>\@glxtr@warn@onexistsordo</code>	7, 11
<code>\@glxtr@org@markboth</code>	262	<code>\@glxtr@warn@undefaction</code>	7, 11
<code>\@glxtr@org@markright</code>	262	<code>\@glxtr@docdeffalse</code>	38
<code>\@glxtr@org@newacronymstyle</code>	93, 94	<code>\@glxtr@entryfmt</code>	22
<code>\@glxtr@org@postdescription</code>	147	<code>\@glxtr@trifhyphenstart</code>	245
<code>\@glxtr@org@see@noindex</code>	110, 111	<code>\@glxtr@indexaliased</code>	66
<code>\@glxtr@org@setacronymstyle</code>	93, 94	<code>\@glxtr@indexcrossrefsfalse</code>	12
<code>\@glxtr@org@printglossary</code>	41, 99	<code>\@glxtr@indexcrossrefstrue</code>	12
<code>\@glxtr@org@warndep</code>	151	<code>\@glxtr@inmark</code>	262
<code>\@glxtr@p@acrlong@</code>	71	<code>\@glxtr@long</code>	70, 162
<code>\@glxtr@p@acrlongpl@</code>	71	<code>\@glxtr@longpl</code>	71, 165, 166
<code>\@glxtr@p@acrshort@</code>	71	<code>\@glxtr@notinmark</code>	262
<code>\@glxtr@p@acrshortpl@</code>	71	<code>\@glxtr@p</code>	76
<code>\@glxtr@p@long@</code>	70	<code>\@glxtr@p@opt</code>	74
<code>\@glxtr@p@longpl@</code>	71	<code>\@glxtr@pl</code>	40, 41
<code>\@glxtr@p@plural@</code>	70	<code>\@glxtr@postloctag</code>	44, 46
<code>\@glxtr@p@short@</code>	70	<code>\@glxtr@preloctag</code>	43, 44, 46
<code>\@glxtr@p@shortpl@</code>	71	<code>\@glxtr@setaliasnoindex</code>	66, 67
<code>\@glxtr@p@text@</code>	70	<code>\@glxtr@short</code>	70, 161
<code>\@glxtr@pagetag</code>	44, 45	<code>\@glxtr@shortpl</code>	71, 164
<code>\@glxtr@pagetag</code>	44, 45	<code>\@glxtr@undeftag</code>	6, 20
<code>\@glxtr@prevunitcount</code>	88	<code>\@gobble</code>	7, 10, 11, 13, 51, 152
<code>\@glxtr@printglossopts</code>	41, 98, 99	<code>\@gobbletwo</code>	151
<code>\@glxtr@provide@addstoragekey</code>	21	<code>\@ifnextchar</code>	68
<code>\@glxtr@provide@storagekey</code>	21	<code>\@ifpackageloaded</code>	
<code>\@glxtr@record</code>	10, 11, 47, 48, 50	... 5, 13, 112, 117, 134, 135, 138, 140, 279	
<code>\@glxtr@recordsee</code>	11	<code>\@ifstar</code>	21, 26, 27, 29, 39, 68, 113, 145
<code>\@glxtr@redef@forglsentries</code>	7, 19	<code>\@ifundefined</code>	279
<code>\@glxtr@redefstyles</code>	17, 18, 279	<code>\@ignored@glossaries</code>	27–30
<code>\@glxtr@reg@glosslist</code>	95–98, 100	<code>\@input</code>	111
<code>\@glxtr@s@longnewglossaryentry</code>	26	<code>\@input@</code>	106
<code>\@glxtr@savepreloctag</code>	44, 45	<code>\@istfilename</code>	95
<code>\@glxtr@setentrycountunsetattr</code>	80	<code>\@makeglossary</code>	95
<code>\@glxtr@setentryunitcountunsetattr</code>		<code>\@mfu@domakefirstuc</code>	146
.....	91, 92	<code>\@mfu@nocaplist</code>	146
<code>\@glxtr@setupshortcuts</code>	16, 19	<code>\@ne</code>	82, 91
<code>\@glxtr@shortcutsval</code>	16, 112	<code>\@newglossaryentry@defcounters</code> ..	80, 88

\@newglossaryentryposthook	9, 10, 21, 34, 63	short-long-user	240
\@newglossaryentryprehook	9, 10, 21, 26, 27, 34, 63	short-nolong	183
\@nil	105, 116	short-nolong-desc	185
\@nnil	141, 142, 144, 150, 152, 153	short-postlong-user	242
\@no@glstrindexaliased	66, 67	\abbreviationsname	13
\@no@makeglossaries	110	\abbrvpluralsuffix	112, 154, 174, 176, 178, 180, 182, 184, 185, 189, 191, 192, 194, 195, 197, 198, 200, 202, 204, 206, 207, 209, 210, 212, 214, 216, 218, 219, 221, 223–225, 227, 229, 231, 233, 235, 237, 238, 241, 244, 247, 248, 252, 255, 257, 260
\@nopostdesc	99	\ABP	14
\@onelevel@sanitize	9, 34, 41, 102, 103, 116	\Abp	14
\@onlypreamble	42, 44, 90, 111, 113, 140, 142, 143, 145	\abp	14
\@org@glossaryentrynumbers	99	\AC	15
\@org@newglossaryentryprehook	26, 27	\Ac	14
\@print@unsrt@glossary	114	\ac	14
\@printgloss@setsort	98, 99	\ACF	15
\@printglossary	41, 114	\Acf	15
\@printunsrt@glossary@handler	114	\acf	14
\@printunsrtglossary	113	\ACFP	15
\@sGlsXtrEnableOnTheFly	39	\Acfp	15
\@secondofthree	52–54, 59–62, 64, 158, 160, 161, 163, 165, 166	\acfp	14
\@secondoftwo	48, 51, 54–62, 65, 70, 94, 99, 151, 158, 159, 161–167, 180, 200, 214, 235, 263, 264	\ACL	15
\@sglsxtr@provide@storagekey	21	\Acl	14
\@thirdofthree	52–54, 59–62, 64, 159, 160, 162, 163, 165, 167	\acl	14
\@thirdoftwo	54–58	\ACLP	15
\@warn@nomakeglossaries	107	\Aclp	15
\@xdy@main@language	106	\aclp	14
\@xdycrossrefhook	33	\ACP	15
\@xdy@language	106	\Acp	14
\@xdy@locationclassorder	33	\acp	14
\\	105	\ACRfullfmt	93
_	108, 109	\Acrfullfmt	93
A			
\AB	14	\acrfullfmt	93
\Ab	14	\ACRfullplfmt	93
\ab	14	\Acrfullplfmt	93
abbreviation styles:		\Acrfullplfmt	93
long-hyphen-postshort-hyphen	250, 251, 253	\acronymentry	93
long-hyphen-short-hyphen	247, 251	\acronymfont	59–62, 73, 93, 94
long-postshort-user	239	\acronymname	13
long-short-user	238	\acronymsort	93
short	183	\acronymtype	13, 92, 93
short-hyphen-long-hyphen	255, 256	\acrpluralsuffix	93, 112
short-hyphen-postlong-hyphen	256, 258	\ACS	15
		\Acs	14
		\acs	14
		\ACSP	15

\Acsp	14	markshortwords	154
\acsp	14	markwords	153, 155, 245, 246, 254
\actualchar	143	nohyper	65
\addtolength	296	nohyperfirst	52–54
\advance	82, 91, 111	noshortplural	154
\AF	14	regular	46, 85, 173, 175–178, 180, 182–188, 190, 192–194, 196–198, 200, 203–206, 208, 209, 211–213, 217, 219–223, 225, 226, 228, 230–234, 237, 243, 244, 246–248, 250, 254, 255, 259, 261
\Af	14	\cGLS	14, 15, 80, 91
\af	14	\cGls	14, 80, 91
\AFP	14	\cglS	14, 80, 91
\Afp	14	\cGLSformat	84
\afp	14	\cGlsformat	83
\AL	14	\cglSformat	83, 85
\Al	14	\cGLSpl	14, 15, 80, 91
\al	14	\cGlspl	14, 80, 91
\ALP	14	\cglSpl	14, 80, 91
\Alp	14	\cGLSplformat	84
\alp	14	\cGlsplformat	84
\AnyTrackedLanguages	279	\cglSplformat	83, 85
\appto	9, 10, 18, 21, 32–36, 63, 67, 80, 88, 115, 140, 150, 152, 153	\char	102
\AS	14	\columnwidth	43
\As	14	\count@	82, 91
\as	14	\csappto	25
\ASP	14	\csdef	21, 24–26, 63, 64, 81, 86, 87, 89, 128, 170–172, 180, 200, 213, 234, 238, 240, 242, 251, 253, 257, 258, 281
\Asp	14	\cseappto	30
\asp	14	\csedef	24, 88
\AtBeginDocument	20, 43, 112	\csgdef	25, 27–30, 38, 44, 81, 82, 87, 89, 90
\AtEndDocument	36, 82, 90, 106, 107	\cslet	24, 26, 27, 99
B		\csletcs	24, 172
babel package	140, 142, 149	\csname ..	6, 7, 21, 28, 29, 34, 42, 46, 48–50, 59–64, 66, 74, 87, 88, 96, 103, 106, 109, 110, 114, 134, 151, 158–167, 172, 173, 295
\begin	103, 108, 114, 281–285	\cspreteto	26
\begingroup	7, 66, 114	\csuse	22, 28, 36, 44, 63, 64, 76, 77, 86–90, 98, 102–104, 113, 115, 116, 128, 139, 147, 148, 170–172, 288–291
\bgroup	26, 99	\csxdef	32, 34, 35, 87, 90, 103
C		\currentglossary	99
\catcode	111	\CurrentOption	18, 280
category attributes:		\CurrentTrackedLanguageTag	112
aposplural	154	\CurrentTrackedTag	279
discardperiod	149	\CustomAbbreviationFields	155, 173, 175, 177–179, 181, 183, 185, 187, 188, 190–193, 195, 198, 200, 202–205,
entrycount	79, 80, 82, 91		
firsttuc	138		
glossdesc	134		
glossdescfont	134		
glossname	135		
glossnamefont	136, 138		
headuc	264		
indexname	141		
indexonlyfirst	67		
insertdots	154		

207, 208, 211, 213, 215–222, 224, 225, 227, 230, 232, 234, 237–240, 242–244, 246–248, 250, 251, 253–256, 258, 259, 261	
D	
<code>\DeclareAcronymList</code>	92
<code>\DeclareOption</code>	5, 280
<code>\DeclareOptionX</code>	5, 18
<code>\def</code>	7–11, 20, 26– 29, 32, 34, 37, 39–41, 43, 46–62, 66, 68, 70–74, 83–85, 92, 96, 98–100, 102–106, 114, 116, 139, 141–146, 150–154, 158– 167, 170, 245, 248, 250, 251, 254, 256, 296
<code>\defglstryfmt</code>	27–30
<code>\define@boolkey</code>	12, 65
<code>\define@choicekey</code> ...	7, 10, 12, 15, 17, 49, 99
<code>\define@key</code> ..	9, 10, 17, 18, 21, 34, 50, 63, 151
<code>\DefineAcronymSynonyms</code>	16
<code>\delimN</code>	45
<code>\delimR</code>	45
<code>\detokenize</code>	39
<code>\dimen@</code>	94, 288–295
<code>\dimen@i</code>	289–291
<code>\dimen@ii</code>	289–291
<code>\dimexpr</code>	43, 287
<code>\disable@keys</code>	13, 20, 38
<code>\do</code> ..	6, 17, 37, 80, 92, 95, 98, 103, 114, 133, 145
<code>\do@glsl@link@checkfirsthyper</code>	22, 47, 48, 50, 59–62, 158–167
<code>\do@glsl@disablehyperinlist</code>	50, 66
doc package	143
<code>\dolistcsloop</code>	23
<code>\DTLifinlist</code>	96, 97, 100
<code>\DTLifint</code>	102
E	
<code>\eappto</code>	9, 17, 27–30, 114, 140, 280
<code>\edef</code> ..	6, 7, 27–30, 33, 35, 36, 49, 50, 65, 66, 70, 86, 87, 89, 95, 96, 100, 102, 104–107, 111, 134–139, 141, 142, 144, 151, 289–291
<code>\eglssetwidest</code>	288–295
<code>\egroup</code>	26, 27, 99
<code>\else</code>	8, 9, 12, 13, 15, 17, 18, 25, 38, 39, 44, 46, 48, 50, 66, 67, 83, 94, 97, 99, 102, 104, 105, 108, 110, 111, 140–142, 144, 150, 152–154, 161– 167, 169, 170, 174, 176, 178–182, 184– 187, 189, 191–199, 201, 203–216, 218– 239, 241, 242, 244–246, 248, 251, 253, 254, 256, 258, 260, 261, 282–286, 296, 297
<code>\emph</code>	215
<code>\empty</code>	104, 105
<code>\encapchar</code>	143
<code>\end</code>	104, 108, 109, 115, 281–285
<code>\end@glstr@display@loc</code>	104
<code>\endcsname</code> 6, 7, 21, 28, 29, 34, 42, 46, 48–50, 59–64, 66, 74, 87, 88, 96, 103, 106, 109, 110, 114, 134, 151, 158–167, 172, 173, 295	
<code>\endgroup</code>	8, 66, 114
entry categories:	
abbreviation	167
general	128, 130
index	132
<code>\epreto</code>	141
<code>\equal</code>	109, 110
etoolbox package	5
<code>\expandafter</code>	18, 21, 22, 32, 33, 36, 37, 39–41, 63, 64, 68, 74, 85, 86, 96–98, 100, 103–105, 114, 116, 117, 134, 137, 138, 140, 143, 144, 150, 153–155, 245
<code>\expandonce</code>	93, 141, 142, 174, 248
F	
<code>\fi</code> ..	7–9, 11–13, 15–17, 19, 25, 32, 35, 36, 38, 39, 42–44, 46, 48–50, 66, 67, 82, 83, 90, 91, 94, 97–100, 102, 104, 105, 107–111, 113, 140–142, 144, 150, 152, 153, 155, 161–167, 169, 170, 174, 176, 178–182, 184–187, 189, 191–199, 201, 203–216, 218–239, 241, 242, 244–246, 248, 251– 254, 256, 258, 260, 261, 282–286, 288–297
first use	298
flag	298
text	298
<code>\firstacronymfont</code>	93, 94
fontspec package	112
<code>\footnote</code>	177
<code>\forall glossaries</code> ..	36, 114, 131–133, 288–294
<code>\forall glsentries</code>	82, 91
<code>\ForEachTrackedDialect</code>	279
<code>\forall glsentries</code>	6, 36, 131–133, 288–294
<code>\forall listcsloop</code>	23, 91, 103
<code>\forall listloop</code>	100, 101, 146
<code>\futurelet</code>	150
G	
<code>\gdef</code>	45, 142, 143
<code>\Genacrfullformat</code>	93
<code>\genacrfullformat</code>	93
<code>\GenericAcronymFields</code>	93

<code>\Genplacrformat</code>	93	<code>\gls@defdocnewglossaryentry</code>	80, 88
<code>\genplacrformat</code>	93	<code>\gls@defglossaryentry</code>	26, 27, 40, 41
<code>\glo@grabfirst</code>	116	<code>\gls@dotocitle</code>	99
<code>\glo@name</code>	136–138	<code>\gls@glossary</code>	34
<code>\gloaliaslabel</code>	69	<code>\gls@level</code>	116
<code>\global</code>	26, 27, 99, 116, 117	<code>\gls@noidxglossary</code>	96
<code>\glolinkprefix</code>	50, 69, 70, 112, 115	<code>\gls@org@glossaryentryfield</code>	99
<code>glossaries package</code>	11, 18, 19, 32–35, 98, 280	<code>\gls@org@glossarysubentryfield</code>	99
<code>glossaries-accsupp package</code>	17, 18, 117	<code>\gls@save@numberlist</code>	43, 44, 46
<code>glossaries-extra package</code>	2	<code>\gls@set@xr@key</code>	34
<code>glossaries-extra-stylemods package</code> .	17, 147, 279	<code>\gls@tmplen</code>	288–294, 296
<code>glossaries.sty package</code>	27	<code>\gls@type</code>	96
<code>\GlossariesExtraWarning</code>		<code>\glsabbrvdefaultfont</code> ...	157, 174, 176, 178, 180, 182, 184, 185, 236, 246, 249, 259
.....	6, 13, 25, 26, 39, 41, 94, 96, 105, 108, 111, 114, 134–139, 145, 146, 172	<code>\glsabbrvemfont</code> .	215–225, 227, 229, 231–235
<code>\GlossariesExtraWarningNoLine</code> .	13, 82, 91	<code>\glsabbrvfont</code>	71, 72, 93, 157, 161, 162, 164, 165, 169, 170, 173–185, 187, 189, 191, 192, 194, 195, 197, 199, 200, 202, 204, 206, 207, 209, 210, 212, 214, 216, 218, 219, 221, 223– 225, 227, 229, 231, 233, 235, 237, 238, 241, 243, 244, 247, 249–252, 255, 257, 260
<code>\GlossariesWarning</code>	44, 96, 98, 101, 170	<code>\glsabbrvhyphenfont</code>	246, 247, 251–258
<code>\GlossariesWarningNoLine</code>	96, 107	<code>\glsabbrvonlyfont</code>	259–261
<code>glossary styles:</code>		<code>\glsabbrvscfont</code>	188–195, 197–200
<code>alttree</code>	286, 287, 295	<code>\glsabbrvsmfont</code>	202–214
<code>inline</code>	286	<code>\glsabbrvuserfont</code>	236–238, 240–244
<code>listdotted</code>	281	<code>\GLSaccessdesc</code>	55
<code>listdottedstyle</code>	281	<code>\Glsaccessdesc</code>	55, 134, 144
<code>sublistdotted</code>	281	<code>\glsaccessdesc</code>	55, 135, 148
<code>glossary-long package</code>	282	<code>\GLSaccessdescplural</code>	55
<code>glossary-longbooktabs package</code>	282	<code>\Glsaccessdescplural</code>	55
<code>glossary-mcols package</code>	280	<code>\glsaccessdescplural</code>	55
<code>glossary-tree package</code>	280	<code>\GLSaccessfirst</code>	53
<code>\glossaryentrynumbers</code>	46, 99, 116, 117	<code>\Glsaccessfirst</code>	52
<code>\glossaryheader</code>	103, 114, 281–285, 296	<code>\glsaccessfirst</code>	52
<code>\glossaryname</code>	98	<code>\GLSaccessfirstplural</code>	54
<code>\glossarypostamble</code>	104, 115	<code>\Glsaccessfirstplural</code>	54
<code>\glossarypreamble</code>	103, 114	<code>\glsaccessfirstplural</code>	53
<code>\glossarysection</code> ...	103, 104, 109, 114, 115	<code>\Glsaccesslong</code>	61, 156, 163, 174, 182, 186, 189, 193, 195–197, 203, 206, 209–211, 216, 218, 223, 226– 231, 238, 239, 247, 249, 252, 253, 255, 260
<code>\glossarytitle</code> 28, 29, 98, 99, 103, 104, 109, 114		<code>\glsaccesslong</code>	61, 156, 162, 163, 174, 176, 178, 179, 181, 182, 184–186, 189, 191, 192, 194–199, 201, 203–216, 218, 220–235, 237, 239, 241, 244, 247, 249, 252, 255, 260
<code>\glossarytoctitle</code>	28, 29, 98, 103, 104, 109, 114		
<code>\glossentry</code>	99, 117, 281–285, 296		
<code>\glossentrydesc</code>	281–287		
<code>\glossentryname</code>	281–285, 296		
<code>\glossentrysymbol</code>	282–287		
<code>\glossxtrsetpopts</code>	147		
<code>\GLS</code>	80, 91		
<code>\Gls</code>	40, 80, 91		
<code>\gls</code>	25, 40, 42, 80, 91, 97, 108		
<code>\gls@assign@desc</code>	26, 27		
<code>\gls@assign@field</code>	9, 10, 21, 63		
<code>\gls@checkseeallowed</code>	38, 96		
<code>\gls@codepage</code>	107		

<code>\Glsaccesslongpl</code>	62, 156, 166, 174, 182, 186, 189, 193, 195–197, 203, 206, 209–211, 216, 218, 223, 226–231, 238, 239, 247, 249, 250, 252, 253, 255, 260, 261
<code>\glsaccesslongpl</code>	62, 156, 166, 167, 174, 176, 178, 179, 181, 182, 184, 186, 187, 189, 191, 192, 194–201, 203, 205–213, 215, 216, 218, 220–224, 226–236, 238, 239, 241, 242, 244, 247, 249, 252, 253, 255, 260
<code>\GLSaccessname</code>	54
<code>\Glsaccessname</code>	54
<code>\glsaccessname</code>	54
<code>\GLSaccessplural</code>	53
<code>\Glsaccessplural</code>	53
<code>\glsaccessplural</code>	53
<code>\Glsaccessshort</code>	59, 161, 169, 176, 179–181, 184, 191, 194, 199, 201, 205, 207, 212–215, 220, 221, 224, 233–235, 241, 244, 252, 257, 258
<code>\glsaccessshort</code>	59, 156, 161, 162, 169, 174, 176, 178–182, 184, 186, 189, 191–194, 196, 197, 199, 201, 203, 204, 206–209, 211–214, 216, 218, 219, 221, 223–226, 228–231, 233, 235, 237–239, 241, 244, 247, 249, 252, 255, 257, 258, 260
<code>\Glsaccessshortpl</code>	60, 165, 170, 176, 179, 181, 184, 191, 194, 199, 201, 205, 208, 212–215, 220, 222, 224, 233–235, 241, 242, 244, 252, 257, 258, 261
<code>\glsaccessshortpl</code>	60, 156, 164, 165, 169, 174, 176, 178–182, 184, 186, 189, 191–194, 196, 197, 199, 201, 203, 204, 206–209, 211–214, 216, 218, 220, 221, 223–226, 228–231, 233–235, 238, 239, 241, 244, 247, 249, 252, 255, 257, 258, 260
<code>\GLSaccesssymbol</code>	56
<code>\Glsaccesssymbol</code>	56, 145
<code>\glsaccesssymbol</code>	56, 144, 148
<code>\GLSaccesssymbolplural</code>	56
<code>\Glsaccesssymbolplural</code>	56
<code>\glsaccesssymbolplural</code>	56
<code>\GLSaccesstext</code>	52
<code>\Glsaccesstext</code>	52
<code>\glsaccesstext</code>	51
<code>\glsacrshortcutstrue</code>	16
<code>\glsacspacemax</code>	94
<code>\glsadd</code>	37, 108
<code>\glsadd options</code>	
thevalue	8
<code>\glsaddstoragekey</code>	35, 36, 128
<code>\glsbackslash</code>	39
<code>\glscapscase</code>	48, 51–62, 64, 158–169
<code>\glscategory</code>	46, 51, 65, 71, 72, 129–131, 134–139, 144, 145, 147, 148, 158–162, 164, 165
<code>\glscategorylabel</code>	65, 151, 153–155, 180, 200, 213, 234, 238, 240, 242, 251, 253, 257, 258
<code>\glsclsebrace</code>	33, 109, 110
<code>\glscurrententrylabel</code>	43–45, 99, 106, 114, 115, 146, 147
<code>\glscurrentfieldvalue</code>	22, 236
<code>\glscustomtext</code>	47, 48, 59–62, 158–167, 169
<code>\glsdefaulttype</code>	6, 13, 25, 98, 107, 114, 115
<code>\glsdescriptionaccessdisplay</code>	121, 122, 134
<code>\glsdescriptionpluralaccessdisplay</code>	122
<code>\glsdescwidth</code>	281, 283–285
<code>\glsdetoklabel</code>	23–27, 30–33, 37–39, 49, 50, 66, 69, 81, 82, 87–91, 96, 99–101, 116, 134, 136–138, 289–291
<code>\glsdisplaynumberlist</code>	97, 100
<code>\glsdohyperlink</code>	70
<code>\glsdohypertarget</code>	99
<code>\glsdoifexists</code>	12, 24, 30, 32, 33, 47, 48, 50, 59–62, 101, 158–167
<code>\glsdoifexistsordo</code>	22, 48
<code>\glsdoifexistsorwarn</code>	12, 134, 135, 144
<code>\glsdoifnoexists</code>	26
<code>\glsdonohyperlink</code>	50, 70
<code>\glsdosanitizesort</code>	97
<code>\glsenableentrycount</code>	80, 82, 90
<code>\glsenableentryunitcount</code>	82, 91
<code>\glsentrycounter</code>	105
<code>\glsentrycurrcount</code>	81, 82, 88
<code>\Glsentrydesc</code>	122, 126, 135
<code>\glsentrydesc</code>	121, 122, 126, 127, 135
<code>\Glsentrydescplural</code>	122, 127
<code>\glsentrydescplural</code>	122, 127
<code>\Glsentryfirst</code>	85, 119, 125
<code>\glsentryfirst</code>	85, 119, 120, 125, 126, 275
<code>\Glsentryfirstplural</code>	86, 120, 126
<code>\glsentryfirstplural</code>	86, 120, 126, 276
<code>\glsentryfmt</code>	27–30
<code>\Glsentryfull</code>	93
<code>\glsentryfull</code>	93
<code>\Glsentryfullpl</code>	93

<code>\glsentryfullpl</code>	93	<code>\glsfirstabbrvfont</code> 93, 156, 173–186, 189, 191, 192, 194,
<code>\glsentryitem</code>	281–285, 296		195, 197, 199, 201, 202, 204, 206, 207,
<code>\Glsentrylong</code>	72, 73, 85, 124, 127		209, 210, 212, 214, 216, 218, 219, 221,
<code>\glsentrylong</code>	72, 73, 85, 124, 127,		223–225, 227, 229, 231, 233, 235, 237,
	180, 200, 214, 234, 240, 242, 256, 276, 277		238, 241, 244, 247, 249, 252, 255, 257, 260
<code>\Glsentrylongpl</code>	73, 86, 124, 128	<code>\glsfirstabbrvhyphenfont</code> 246, 247, 251, 252, 254–258
<code>\glsentrylongpl</code>	73, 74, 86, 124, 128, 277	<code>\glsfirstabbrvonlyfont</code>	260, 261
<code>\Glsentryname</code>	118, 125, 136–139	<code>\glsfirstabbrvscfont</code>	188–201
<code>\glsentryname</code> ...	117, 118, 125, 141, 288–295	<code>\glsfirstabbrvsmfont</code>	202–215
<code>\glsentrynumberlist</code>	97, 102, 293–295	<code>\glsfirstabbrvuserfont</code>	237–244
<code>\Glsentryplural</code>	119, 125	<code>\glsfirstaccessdisplay</code>	119, 120
<code>\glsentryplural</code>	119, 125, 275	<code>\glsfirstlongdefaultfont</code> 174, 176, 182, 184, 185, 188–198, 202–
<code>\glsentryprevcount</code>	81, 82, 88		211, 215–217, 219, 220, 223–227, 229, 230
<code>\glsentryprevmaxcount</code>	89	<code>\glsfirstlonggemfont</code> 217, 218, 221, 222, 227, 228, 230–232
<code>\glsentryprevtotalcount</code>	89	<code>\glsfirstlongfont</code>	156, 173–178,
<code>\Glsentryshort</code>	72, 73, 123, 127		180, 182, 184–187, 189, 191, 192, 194,
<code>\glsentryshort</code>		195, 197, 199, 201, 203, 204, 206, 207,
	71–73, 94, 123, 127, 238, 240, 251, 273		209, 210, 212, 214, 216, 218, 219, 221,
<code>\Glsentryshortpl</code>	72, 73, 123, 127		223–225, 227, 229, 231, 233, 235, 237,
<code>\glsentryshortpl</code> 72, 73, 123, 124, 127, 273, 274		239, 241, 244, 247, 249, 252, 255, 257, 260
<code>\Glsentrysymbol</code>	121, 126	<code>\glsfirstlongfootnotefont</code> 178–181, 198–201, 212–215, 232–236
<code>\glsentrysymbol</code>	120, 121, 126, 292, 293	<code>\glsfirstlonghyphenfont</code>	246–257
<code>\Glsentrysymbolplural</code>	121, 126	<code>\glsfirstlongonlyfont</code>	259–261
<code>\glsentrysymbolplural</code>	121, 126	<code>\glsfirstlonguserfont</code>	237–244
<code>\Glsentrytext</code>	118, 125	<code>\GLSfirstplural</code>	269
<code>\glsentrytext</code>	70, 118, 125, 274	<code>\Glsfirstplural</code>	269
<code>\Glsentryuseri</code>	57	<code>\glsfirstplural</code>	269
<code>\glsentryuseri</code>	57	<code>\glsfirstpluralaccessdisplay</code>	120
<code>\Glsentryuserii</code>	57	<code>\glsforeachincategory</code>	170
<code>\glsentryuserii</code>	57	<code>\glsgenentryfmt</code>	46
<code>\Glsentryuseriii</code>	57	<code>\glsgetattribute</code> 69, 82, 87, 88, 106, 134–140	
<code>\glsentryuseriii</code>	57	<code>\glsgetcategoryattribute</code>	129
<code>\Glsentryuseriv</code>	58	<code>\glsgetwidestname</code>	287
<code>\glsentryuseriv</code>	58	<code>\glsgroupheading</code>	117, 281–285, 296
<code>\Glsentryuserv</code>	58	<code>\glsgroupskip</code>	117, 282–286, 297
<code>\glsentryuserv</code>	58	<code>\glshasattribute</code> 69, 82, 87, 89, 91, 106, 134–140, 173,
<code>\Glsentryuservi</code>	58		175–178, 180, 183, 185, 187–190, 192,
<code>\glsentryuservi</code>	58		198, 200, 202–205, 212, 214, 216, 217,
<code>\glsfieldfetch</code>	69		219–222, 228, 232–234, 237, 238, 240–
<code>\glsfieldxdef</code>	133		244, 246–248, 250, 252–255, 257, 259, 261
<code>\glsfindwidesttoplevelname</code>	288	<code>\glshascategoryattribute</code>	129
<code>\GLSfirst</code>	268	<code>\glshyperlink</code>	70
<code>\Glsfirst</code>	268	<code>\glshypernavsep</code>	103
<code>\glsfirst</code>	268		
<code>\glsfirstabbrvdefaultfont</code>	157, 174, 176, 178, 180, 182, 184, 185, 249		
<code>\glsfirstabbrvemfont</code>	215–235		

<code>\glshypernumber</code>	106, 140	210, 212, 214, 216, 218, 219, 221, 223–
<code>\glsifattribute</code>	49, 52,	225, 227, 229, 231, 233, 235, 237, 239,
	65, 67, 75, 132, 134–137, 146, 149, 264–272	241, 244, 247, 249, 252, 255, 257, 260, 261
<code>\glsifcategory</code>	131	<code>\glslongfootnotefont</code>
<code>\glsifcategoryattribute</code> .	65, 130, 153–155	177, 178, 180, 199, 201, 212, 214, 233, 235
<code>\glsifnotregular</code>	51	<code>\glslonghyphenfont</code> .
<code>\glsifnotregularcategory</code>	131	246–252, 254, 255, 257
<code>\glsifplural</code> 48, 51, 53–56, 59–62, 149, 158–168		<code>\glslongonlyfont</code>
<code>\glsifregular</code>	46, 51, 85, 86	259, 260
<code>\glsifregularcategory</code>	131	<code>\glslongpltok</code>
<code>\glsifusetranslator</code>	28	155, 173,
<code>\glsignore</code>	45	175, 177, 178, 185, 187, 188, 190, 191,
<code>\glsinlinedescformat</code>	286	195, 198, 202–205, 208, 212, 216–222,
<code>\glsinlinesubdescformat</code>	286	225, 227, 230, 232, 237, 238, 240, 242–
<code>\glsinsert</code>	48,	244, 246–248, 250, 251, 253–255, 259, 261
	51, 59–62, 158–169, 245, 251–253, 257–259	<code>\glslongpluralaccessdisplay</code>
<code>\glskeylisttok</code>	92, 93, 153, 155	124
<code>\glslabel</code>	30, 46, 49, 50, 65,	<code>\glslongtok</code> ...
	66, 69, 70, 94, 148, 167–169, 180, 200,	92, 93, 153–155, 173–179,
	214, 234, 238, 240, 242, 251–253, 257–259	181, 183, 185, 187–193, 195, 198, 200,
<code>\glslabeltok</code>	92,	202–208, 212, 213, 215–222, 224, 225,
	153, 155, 173–178, 180, 181, 183, 185,	227, 230, 232, 234, 237–240, 242–244,
	187–192, 194, 195, 198, 200, 202–208,	246–248, 250, 251, 253–255, 257, 259, 261
	212, 214–225, 227–230, 232–234, 237,	<code>\glslonguserfont</code>
	238, 240–248, 250, 252–255, 257, 259–261	237–244
<code>\glsletentryfield</code>	141	<code>\glsnameaccessdisplay</code> ..
<code>\glslink</code>	93	117, 118, 136–138
<code>\glslink options</code>		<code>\glsnamefont</code>
format	139	136–139
hyper	262	<code>\glsnavhyperlink</code>
noindex	7, 65, 262	103
wrgloss	49	<code>\glsnextpages</code>
<code>\glslinkcheckfirsthyperhook</code>	65	99
<code>\glslinkpostsetkeys</code>	50	<code>\glsnoidxdisplayloc</code>
<code>\glslinkvar</code>	68	101
<code>\glslistdottedwidth</code>	281	<code>\glsnoidxdisplaylocclsthandler</code> ...
<code>\glslocalunset</code>	48	100
<code>\glslongaccessdisplay</code>	124	<code>\glsnoidxloclist</code>
<code>\glslongdefaultfont</code>		101, 116, 117
	. 157, 174, 176, 177, 182, 184, 185, 189,	<code>\glsnoidxnumberlistloophandler</code> ...
	191, 192, 194, 195, 197, 203, 204, 206–	101
	210, 216, 219, 223–226, 229, 236, 246, 259	<code>\glsnonnextpages</code>
<code>\glslongemfont</code> 215, 217, 218, 221, 227, 230, 231		99
<code>\glslongfont</code>	72, 73, 157,	<code>\glsnonnumberlistfalse</code>
	162, 163, 166, 167, 174, 176, 178, 180,	43
	182, 184–187, 189, 191, 192, 194, 195,	<code>\glsnonnumberlisttrue</code>
	197, 199, 201, 203, 204, 206, 207, 209,	44
		<code>\glsnumberlistloop</code>
		97
		<code>\glsnumlistlastsep</code>
		100
		<code>\glsnumlistsep</code>
		100
		<code>\glsopenbrace</code>
		33, 109, 110
		<code>\glsorder</code>
		95
		<code>\glspagelistwidth</code>
		281, 283–285
		<code>\glspar</code>
		115
		<code>\GLSpl</code>
		80, 91
		<code>\Glspl</code>
		41, 80, 91
		<code>\glspl</code>
		40, 80, 91
		<code>\GLSplplural</code>
		267
		<code>\Glsplural</code>
		267, 268
		<code>\glsplural</code>
		267
		<code>\glspluralaccessdisplay</code>
		119
		<code>\glspluralsuffix</code>
		112, 153, 157
		<code>\glspostdescription</code>
		147, 281–287
		<code>\glspostinline</code>
		286
		<code>\glspostlinkhook</code> .
		47–49, 59–63, 74, 158–167
		<code>\glsprestandardsort</code>
		97

<code>\glsresetentrylist</code>	103, 114	<code>\glsxtr@addloccllistfield</code>	11
<code>\glssee</code>	34–36	<code>\glsxtr@addunused</code>	37
<code>\glsseeformat</code>	32, 33, 96, 101	<code>\glsxtr@applyabbrvfmt</code>	167
<code>\glsseelist</code>	33	<code>\glsxtr@applyabbrvstyle</code>	151, 153, 170
<code>\glssetabbrvfmt</code>	46, 51, 71, 72, 134–139, 144, 145, 158–162, 164, 165	<code>\glsxtr@counterrecord</code>	113
<code>\glssetattribute</code> 174–178, 180, 181, 183, 185, 187–190, 192, 194, 195, 198, 200, 202–208, 212, 214, 216, 217, 219–225, 227, 229, 230, 232–234, 237, 238, 240– 243, 245–248, 250, 252–255, 257, 259–261		<code>\glsxtr@doption</code>	5, 13, 18, 19
<code>\glssetcategoryattribute</code>		<code>\glsxtr@fields</code>	112
..... 80, 92, 94, 129, 130, 132, 133, 145		<code>\glsxtr@headentry@p</code>	75, 76
<code>\glssetnoexpandfield</code>	9, 10	<code>\glsxtr@ifnextpunc</code>	150
<code>\glssettoctitle</code>	99	<code>\glsxtr@ifpunctoken</code>	150
<code>\glsshortaccessdisplay</code>	123	<code>\glsxtr@indexonly@saveentrycounter</code>	10, 11, 20
<code>\glsshortpltok</code>		<code>\glsxtr@keylist</code>	39–41
.... 155, 173, 175, 177–181, 183, 189– 193, 198, 200, 202–207, 212, 213, 216– 222, 224, 232, 234, 237, 238, 240, 242– 244, 246, 247, 251, 253–256, 258, 259, 261		<code>\glsxtr@langtag</code>	112
<code>\glsshortpluralaccessdisplay</code> ..	123, 124	<code>\glsxtr@linkprefix</code>	112
<code>\glsshorttok</code>		<code>\glsxtr@makeglossaries</code>	95
.. 92, 93, 153–155, 173–179, 181, 183, 187, 188, 190–193, 195, 198, 200, 202– 208, 211–213, 215, 217–222, 224, 225, 227, 232, 234, 237, 238, 240, 242–244, 246, 247, 250, 251, 253–256, 258, 259, 261		<code>\glsxtr@newabbreviation</code>	93, 153
<code>\glssubentryitem</code>	281–286, 296	<code>\glsxtr@next</code>	150
<code>\glssymbolaccessdisplay</code>	120, 121	<code>\glsxtr@org@getgrouptitle</code>	102
<code>\glssymbolpluralaccessdisplay</code>	121	<code>\glsxtr@org@newignoredglossary</code>	27
<code>\glstarget</code>	281–286, 296	<code>\glsxtr@org@makenoidxglossaries</code>	37
<code>\GLStext</code>	266, 267	<code>\glsxtr@pluralsuffixes</code>	112
<code>\Glstext</code>	267	<code>\glsxtr@provideignoredglossary</code>	29
<code>\glstext</code>	266	<code>\glsxtr@punclist</code>	150
<code>\glstextaccessdisplay</code>	118	<code>\glsxtr@record</code>	8
<code>\glstextformat</code>	49, 50	<code>\glsxtr@recordsee</code>	9
<code>\glstextup</code>	188	<code>\glsxtr@resource</code>	110, 112
<code>\glstreeindent</code>	295, 296	<code>\glsxtr@s@newignoredglossary</code>	27
<code>\glstreenamebox</code>	296	<code>\glsxtr@s@provideignoredglossary</code> ...	29
<code>\glstreenamefmt</code>	287–296	<code>\glsxtr@saveentrycounter</code>	8, 9, 66
<code>\GlstrLetField</code>	24	<code>\glsxtr@setup@record</code>	10, 11, 19, 20
<code>\glstype</code>	48, 49, 59–63, 158–167	<code>\glsxtr@shortcutsval</code>	112
<code>\glunset</code>	37, 48, 83, 84	<code>\glsxtr@texencoding</code>	112
<code>\glswrite</code>	33, 95	<code>\glsxtr@usesee</code>	32
<code>\glswriteentry</code>	8	<code>\glsxtr@warnonexistsordo</code>	7, 10, 11, 31
<code>\Glsxtr</code>	41	<code>\glsxtr@writefields</code>	110
<code>\glsxtr</code>	41	<code>\glsxtrabbrvfootnote</code>	
<code>\glsxtr@do@wrglossary</code>	8, 10, 11 178–180, 198–200, 212–214, 232–234	
		<code>\glsxtrabbrvpluralsuffix</code>	112,
		157, 174, 176, 178, 180, 182, 184, 185, 188, 202, 215, 237, 246, 248, 252, 257, 259	
		<code>\glsxtrabbrvtype</code>	13, 155
		<code>\glsxtraddallcrossrefs</code>	36
		<code>\glsxtralias</code>	66
		<code>\glsxtrAltTreeIndent</code>	287
		<code>\glsxtrAlttreeInit</code>	295
		<code>\glsxtrAltTreePar</code>	286
		<code>\glsxtrAltTreeSetHangIndent</code> ...	287, 296
		<code>\glsxtrAltTreeSetSubHangIndent</code>	296

<code>\glxtralttreeSubSymbolDescLocation</code>	296	225, 226, 228, 230, 232, 233, 235, 238,
<code>\glxtralttreeSymbolDescLocation</code>	..	239, 241, 244, 247, 249, 252, 255, 257, 260
.....	287, 296	
<code>\glxtrassignfieldfont</code>	51–58	<code>\glxtrfullformat</code>
<code>\glxtrautoindex</code>	140	. 156, 169, 171, 172, 174, 176, 178, 180,
<code>\glxtrautoindexassignsort</code>	141	182, 184, 186, 189, 191, 193, 194, 196,
<code>\glxtrautoindexentry</code>	141	197, 199, 201, 203, 204, 206, 208, 209,
<code>\glxtrcat</code>	40, 41	211, 212, 214, 216, 218, 219, 221, 223,
<code>\glxtrchecknohyperfirst</code>	52–54	225, 226, 228, 230, 232, 233, 235, 237,
<code>\glxtrComputeTreeIndent</code>	296	239, 241, 244, 247, 249, 252, 255, 257, 260
<code>\glxtrComputeTreeSubIndent</code>	296	<code>\GLSxtrfullpl</code>
<code>\Glsxtrdefaultsubsequentfmt</code>	169, 171	14, 15, 272
<code>\glxtrdefaultsubsequentfmt</code>	169, 171	<code>\Glsxtrfullpl</code>
<code>\Glsxtrdefaultsubsequentplfmt</code>	170, 171	14, 15, 272, 273
<code>\glxtrdefaultsubsequentplfmt</code>	169, 171	<code>\glxtrfullpl</code>
<code>\GlsXtrDefineAbbreviationShortcuts</code>	16	14, 272
<code>\GlsXtrDefineAcShortcuts</code>	16	<code>\Glsxtrfullplformat</code>
<code>\GlsXtrDefineOtherShortcuts</code>	16 156, 168, 171, 172, 174, 176, 179,
<code>\glxtrdiscardperiod</code>	148	181, 182, 184, 186, 189, 191, 193, 194,
<code>\glxtrdisplayendloc</code>	104	196, 198, 199, 201, 203, 205, 207, 208,
<code>\glxtrdisplayendloohook</code>	105	210–212, 214, 216, 218, 220, 222, 223,
<code>\glxtrdisplaysingleloc</code>	104, 105	225, 227, 228, 230, 232, 233, 235, 238,
<code>\glxtrdisplaystartloc</code>	104	239, 241, 244, 247, 250, 252, 255, 258, 260
<code>\glxtrdoautoindexname</code>	67, 68, 139	<code>\glxtrfullplformat</code>
<code>\glxtrdopostpunc</code>	180, 200, 214, 234 168, 171, 172, 174, 176, 178,
<code>\glxtrdowrglossaryhook</code>	67	180, 182, 184, 186, 189, 191, 193, 194,
<code>\glxtremsuffix</code>	216, 218,	196, 198, 199, 201, 203, 204, 206, 208,
219, 221, 223–225, 227, 229, 231, 233, 235		210–212, 214, 216, 218, 220, 221, 223,
<code>\GlsXtrEnableEntryCounting</code>	92	225, 226, 228, 230, 232, 233, 235, 237,
<code>\GlsXtrEnableEntryUnitCounting</code>	80	239, 241, 244, 247, 249, 252, 255, 257, 260
<code>\GlsXtrEnableOnTheFly</code>	39, 41, 42	<code>\glxtrfullsep</code> 156, 173–177, 179, 181, 182,
<code>\glxtrfieldlistgadd</code>	113	184, 186, 188–194, 196, 197, 199, 201–
<code>\glxtrfieldtitlecase</code>	134–137	209, 211, 213–224, 226, 228–231, 233–
<code>\glxtrfieldtitlecasecs</code>	134	236, 246, 247, 249, 251, 254–256, 260, 261
<code>\glxtrfieldxifinlist</code>	115	<code>\glxtrrgenabbrvfmt</code>
<code>\glxtrfirstscfont</code>	188	46
<code>\glxtrfirstsmfont</code>	202	<code>\glxtrgetgrouptitle</code>
<code>\GlsXtrFmtDefaultOptions</code>	22	103
<code>\GlsXtrFmtField</code>	22	<code>\Glsxtrheadfirst</code>
<code>\GlsXtrFormatLocationList</code>	44, 46, 293–295	263
<code>\GLSxtrfull</code>	14, 15, 271, 272	<code>\glxtrheadfirst</code>
<code>\Glsxtrfull</code>	14, 15, 272	263
<code>\glxtrfull</code>	14, 271	<code>\Glsxtrheadfirstplural</code>
<code>\Glsxtrfullformat</code>		264
.... 156, 169, 171, 172, 174, 176, 179,		<code>\glxtrheadfull</code>
180, 182, 184, 186, 189, 191, 193, 194,		264
196, 198, 199, 201, 203, 205, 207, 208,		<code>\glxtrheadfull</code>
210–212, 214, 216, 218, 220, 221, 223,		264
		<code>\Glsxtrheadfullpl</code>
		264
		<code>\glxtrheadfullpl</code>
		264
		<code>\Glsxtrheadlong</code>
		264
		<code>\glxtrheadlong</code>
		264
		<code>\Glsxtrheadlongpl</code>
		264
		<code>\glxtrheadlongpl</code>
		264
		<code>\Glsxtrheadplural</code>
		263
		<code>\glxtrheadplural</code>
		263
		<code>\Glsxtrheadshort</code>
		263
		<code>\glxtrheadshort</code>
		263
		<code>\Glsxtrheadshortpl</code>
		263

<code>\glxtrheadshortpl</code>	263	<code>\glxtrlonghyphen</code>	252
<code>\Glsxtrheadtext</code>	263	<code>\glxtrlonghyphennoshort</code>	249, 250
<code>\glxtrheadtext</code>	263	<code>\glxtrlonghyphenshort</code>	247
<code>\glxtrhyphensuffix</code>	247, 255	<code>\GLSxtrlongpl</code>	14, 15, 270, 271
<code>\glxtrtrifcounttrigger</code>	83, 84	<code>\Glsxtrlongpl</code>	14, 15, 271
<code>\glxtrtrifemptyglossary</code>	104, 109, 114	<code>\glxtrlongpl</code>	14, 270
<code>\glxtrtrifhyphenstart</code>	245, 248, 250, 251, 254, 256	<code>\glxtrlongshortdescname</code>	175, 190, 203, 217, 218, 242, 247, 253
<code>\glxtrtrifindexing</code>	67	<code>\glxtrlongshortdescsort</code>	175, 190, 203, 217, 218, 242, 247, 253
<code>\glxtrtrifinmark</code>	76–78, 263, 264	<code>\glxtrmarkhook</code>	262
<code>\glxtrtrifnextpunc</code>	150, 151	<code>\glxtrnewabbrevpresetkeyhook</code>	154
<code>\glxtrtrifperiod</code>	149	<code>\glxtrnewnumber</code>	15
<code>\glxtrtrifwasfirstuse</code>	51–54, 59–62, 65, 94, 148, 149, 158, 161–167, 180, 200, 213, 214, 234, 235, 238, 240, 242, 251, 253, 257, 258	<code>\glxtrnewsymbol</code>	15
<code>\glxtrindexaliased</code>	66	<code>\glxtrNoGlossaryWarning</code>	17, 106
<code>\glxtrindexseealso</code>	35, 36	<code>\GlsXtrNoGlsWarningAutoMake</code>	110
<code>\glxtrtrinitwrgloss</code>	49	<code>\GlsXtrNoGlsWarningBuildInfo</code>	110
<code>\glxtrtrinitwrglossbeforefalse</code>	49	<code>\GlsXtrNoGlsWarningCheckFile</code>	110
<code>\glxtrtrinitwrglossbeforetrue</code>	49	<code>\GlsXtrNoGlsWarningEmptyMain</code> ..	109, 110
<code>\Glsxtrinlinenfullformat</code> 157, 158, 171, 172, 179, 181, 182, 184, 186, 193, 194, 196, 197, 199, 201, 206, 207, 209, 211, 213, 215, 223, 224, 226, 228, 230, 231, 234, 235, 239, 241, 249, 253, 258, 260, 278		<code>\GlsXtrNoGlsWarningEmptyNotMain</code> ...	109
<code>\glxtrinlinenfullformat</code>	156, 158, 159, 171, 172, 179, 181, 182, 184, 186, 192, 194, 195, 197, 199, 201, 206, 207, 209, 211, 213, 214, 223, 224, 226, 228, 229, 231, 233, 235, 239, 241, 249, 252, 258, 260, 277	<code>\GlsXtrNoGlsWarningEmptyStart</code>	109
<code>\Glsxtrinlinenfullplformat</code> 157, 160, 171, 172, 179, 181, 182, 184, 186, 193, 194, 196, 197, 199, 201, 206, 208, 209, 211, 213, 215, 223, 224, 226, 228, 230, 231, 234, 235, 239, 242, 249, 253, 258, 261, 278		<code>\GlsXtrNoGlsWarningHead</code>	109
<code>\glxtrinlinenfullplformat</code>	156, 157, 159, 160, 171, 172, 179, 181, 182, 184, 186, 192, 194, 196, 197, 199, 201, 206, 207, 209, 211, 213, 214, 223, 224, 226, 228, 229, 231, 234, 235, 239, 241, 249, 253, 258, 260, 278	<code>\GlsXtrNoGlsWarningMisMatch</code>	110
<code>\glxtrinsertinsidefalse</code>	173	<code>\GlsXtrNoGlsWarningNoOut</code>	110
<code>\glxtrlocationhyperlink</code>	105	<code>\GlsXtrNoGlsWarningTail</code>	110
<code>\glxtrlocrangefmt</code>	104, 105	<code>\glxtronlydescname</code>	261
<code>\GLSxtrlong</code>	14, 15, 269, 270	<code>\glxtronlydescsort</code>	261
<code>\Glsxtrlong</code>	14, 270	<code>\glxtronlysuffix</code>	260
<code>\glxtrlong</code>	14, 269, 270	<code>\glxtrorg@ifKV@glslink@hyper</code>	47
		<code>\glxtrorglong</code>	153, 174, 248
		<code>\glxtrorgshort</code>	153, 174
		<code>\GLSxtrp</code>	75
		<code>\Glsxtrp</code>	75
		<code>\glxtrp</code>	74, 76, 77
		<code>\glxtrparen</code>	156, 173–177, 179, 181, 182, 184, 186, 188–194, 196, 197, 199– 209, 211, 213–224, 226, 228–231, 233– 236, 246, 247, 249, 251, 254–256, 260, 261
		<code>\Glsxtrpl</code>	41
		<code>\glxtrpl</code>	41
		<code>\glxtrpostdescription</code>	132, 147, 286
		<code>\glxtrpostthyphenlong</code>	257, 258
		<code>\glxtrpostthyphenshort</code>	251, 253
		<code>\glxtrpostthyphensubsequent</code>	252, 253, 257, 259
		<code>\glxtrpostlink</code>	148
		<code>\glxtrpostlinkendsentence</code>	148
		<code>\glxtrpostlinkhook</code>	148
		<code>\glxtrpostlocalreset</code>	79, 81, 89

<code>\glxtrpostlocalunset</code>	79, 81, 89	<code>\glxtrsubsequentfmt</code>	
<code>\glxtrpostlongdescription</code>	27	168, 171, 185, 195, 197,
<code>\glxtrpostnamehook</code>	137–139	209, 210, 225, 227, 229, 231, 249, 252, 257	
<code>\GlsXtrPostNewAbbreviation</code>		<code>\Glsxtrsubsequentplfmt</code>	
.....	155, 171, 173, 175–	167, 171, 186, 195, 197,
178, 180, 181, 183, 185, 187–190, 192,		209, 210, 226, 227, 229, 231, 249, 252, 257	
193, 195, 198, 200, 202–208, 212, 213,		<code>\glxtrsubsequentplfmt</code>	
216, 217, 219–222, 224, 225, 227, 228,		167, 168, 171, 186, 195, 197,
230, 232–234, 237, 238, 240, 242–244,		209, 210, 226, 227, 229, 231, 249, 252, 257	
246–248, 250, 251, 253–255, 257–259, 261		<code>\glxtrsupplocationurl</code>	105, 106
<code>\glxtrpostreset</code>	79, 81, 89	<code>\glxtrtagfont</code>	146
<code>\glxtrpostunset</code>	79, 81, 89	<code>\Glsxtrtitlefirst</code>	263, 264, 275, 276
<code>\glxtrprotectlinks</code>	69, 70	<code>\glxtrtitlefirst</code>	263, 264, 275
<code>\GlsXtrRecordCounter</code>	9	<code>\Glsxtrtitlefirstplural</code>	263, 264, 276
<code>\glxtrregularfont</code>	46, 51	<code>\glxtrtitlefirstplural</code>	263, 264, 276
<code>\glxtrresourcecount</code>	111	<code>\Glsxtrtitlefull</code>	263, 264, 278
<code>\glxtrresourcefile</code>	111	<code>\glxtrtitlefull</code>	263, 264, 277, 278
<code>\glxtrrestoremarkhook</code>	262	<code>\Glsxtrtitlefullpl</code>	263, 264, 278
<code>\glxtrscfont</code>	188	<code>\glxtrtitlefullpl</code>	263, 264, 278
<code>\glxtrscsuffix</code>		<code>\Glsxtrtitlelong</code>	263, 264, 277
....	189, 191, 192, 194, 195, 197, 198, 200	<code>\glxtrtitlelong</code>	263, 264, 276
<code>\GlsXtrSetActualChar</code>	143	<code>\Glsxtrtitlelongpl</code>	263, 264, 277
<code>\glxtrsetaliasnoindex</code>	10, 11, 66	<code>\glxtrtitlelongpl</code>	263, 264, 277
<code>\GlsXtrSetEncapChar</code>	143	<code>\Glsxtrtitleplural</code>	263, 264, 275
<code>\GlsXtrSetEscChar</code>	143	<code>\glxtrtitleplural</code>	263, 264, 275
<code>\glxtrsetfieldifexists</code>	24, 25	<code>\Glsxtrtitleshort</code>	263, 264, 273, 274
<code>\GlsXtrSetLevelChar</code>	143	<code>\glxtrtitleshort</code>	263, 264, 273
<code>\glxtrsetpopts</code>	74	<code>\Glsxtrtitleshortpl</code>	263, 264, 274
<code>\glxtrsetupfulldefs</code>		<code>\glxtrtitleshortpl</code>	263, 264, 273
.....	158–160, 180, 200, 214, 235	<code>\Glsxtrtitletext</code>	263, 264, 274
<code>\GLSxtrshort</code>	14, 15, 78, 265	<code>\glxtrtitletext</code>	263, 264, 274
<code>\Glsxtrshort</code>	14, 265, 266	<code>\glxtrtreetopindent</code>	287, 295
<code>\glxtrshort</code>	14, 265	<code>\glxtrundefaction</code> 7, 10, 11, 20, 27, 28, 30, 31	
<code>\glxtrshortdescname</code> ...	183, 193, 207, 224	<code>\glxtrundeftag</code>	20, 101
<code>\glxtrshorthyphen</code>	257, 258	<code>\glxtrunsrtdo</code>	115
<code>\glxtrshorthyphenlong</code>	255	<code>\GlsXtrUseAbbrStyleFmts</code>	175,
<code>\glxtrshortlongdescname</code>		177, 183, 185, 187, 188, 190, 192, 204,	
....	177, 191, 205, 220, 222, 244, 255, 258	205, 217, 219, 220, 222, 229, 232, 240,	
<code>\glxtrshortlongdescsort</code>		242, 243, 245, 248, 250, 254, 256, 259, 261	
....	177, 191, 205, 220, 222, 244, 255, 258	<code>\GlsXtrUseAbbrStyleSetup</code>	
<code>\GLSxtrshortpl</code>	14, 15, 265, 266	183, 185, 187, 188, 196, 210, 228, 229, 232	
<code>\Glsxtrshortpl</code>	14, 266	<code>\glxtruserfield</code>	236
<code>\glxtrshortpl</code>	14, 265	<code>\glxtruserparen</code>	237–244
<code>\glxtrsmfont</code>	202	<code>\glxtrusersuffix</code>	237, 238, 241, 244
<code>\glxtrsmsuffix</code>		<code>\glxtruseseealsoformat</code>	33
....	202, 204, 206, 207, 209, 210, 212, 214	<code>\glxtruseseeformat</code>	32
<code>\Glsxtrsubsequentfmt</code>		<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	151, 172
.....	168, 171, 186, 195, 197,	<code>\GlsXtrWarning</code>	40, 41
209, 210, 226, 227, 229, 231, 249, 252, 257		<code>\glxtrword</code>	153

<code>\glxtrwordsep</code>	153, 245, 248, 250, 251, 254, 256	<code>\ifglshasparent</code>	116, 288–291
H		<code>\ifglshasshort</code>	46, 51
<code>\hangindent</code>	287, 295	<code>\ifglshassymbol</code>	148, 287
<code>\hbox</code>	281	<code>\ifglsindeonlyfirst</code>	67
<code>\hfill</code>	281	<code>\ifglsnogroupskip</code>	282–286, 297
<code>\href</code>	69	<code>\ifglssonumberlist</code>	46
<code>\hsize</code>	42, 43	<code>\ifglssanitizesort</code>	97
<code>\hss</code>	281	<code>\ifglssubentrycounter</code>	25
<code>\hyperlink</code>	69, 105	<code>\ifglssused</code>	36, 37, 65, 67, 82, 91, 94, 167, 288, 289, 291, 293, 294
<code>\hyperpage</code>	140	<code>\ifglsxindy</code>	106, 108
<code>\hyperref</code>	69, 105	<code>\ifglxtrinitwrglossbefore</code>	49, 50
<code>hyperref package</code>	70, 140, 262, 273	<code>\ifglxtrininsertinside</code>	161–167, 169, 170, 174, 176, 178–182, 184–187, 189, 191– 199, 201, 203–216, 218–239, 241, 242, 244, 246, 248, 251–254, 256, 258, 260, 261
I		<code>\ifHy@hyperindex</code>	140
<code>\if</code>	39	<code>\ifinlistcs</code>	23, 38
<code>\if@glxtr@autoseeindex</code>	18, 19, 32, 34	<code>\ifKV@glslink@hyper</code>	47, 49, 50
<code>\if@glxtr@format@override</code>	140	<code>\ifKV@glslink@local</code>	48
<code>\if@glxtr@docdefrestricted</code>	37	<code>\ifKV@glslink@noindex</code>	8, 9, 66, 67
<code>\if@glxtr@indexcrossrefs</code>	12, 36	<code>\ifnum</code>	11, 12, 82, 90, 91, 102, 111, 296
<code>\ifblank</code>	17, 21, 40, 41, 95	<code>\ifthenelse</code>	109, 110
<code>\ifcase</code>	7, 10, 16, 17, 38, 49, 99	<code>\IfTrackedLanguageFileExists</code>	279
<code>\ifcsdef</code>	7, 20, 25, 27–30, 63, 64, 74–78, 86, 98, 102, 103, 116, 134–139, 148, 151, 167, 171, 281–285	<code>\ifundef</code>	70, 95, 145–147
<code>\ifcsstring</code>	20, 130, 170	<code>\ifx</code>	42, 43, 98, 99, 104, 105, 113, 141, 142, 144, 150, 152–154, 245, 297
<code>\ifcsundef</code>	25–30, 38, 42, 44, 70, 81, 86–90, 102, 103, 106, 115, 130, 170–173, 280, 288, 295	<code>\immediate</code>	82, 90, 91, 106, 107, 112
<code>\ifcsvoid</code>	36, 129	<code>\index</code>	141
<code>\ifdef</code>	11, 15, 18, 19, 22, 31, 33, 34, 42, 43, 65, 75, 77, 78, 98, 100, 101, 112, 132, 133, 143, 147, 236, 273–278, 281, 286	<code>\indexspace</code>	297
<code>\ifdefempty</code>	7, 8, 27–29, 32, 33, 50, 80, 92, 95, 98, 105, 114, 117, 145, 153, 167	<code>\input</code>	279
<code>\ifdefequal</code>	110, 116	<code>\inputencodingname</code>	112
<code>\ifdefstring</code>	6, 140, 146	<code>\istfilename</code>	95
<code>\ifdefvoid</code>	32, 34–37, 69, 86, 102, 105, 116, 117	<code>\item</code>	108, 109, 281
<code>\ifdim</code>	42, 43, 94, 288–295	J	
<code>\IfFileExists</code>	17, 106, 109, 110, 112, 280	<code>\jobname</code>	106, 108–112
<code>\ifglossaryexists</code>	31	K	
<code>\ifglssacronym</code>	13, 109	<code>\key@ifundefined</code>	9, 10, 20, 21, 63, 114, 116
<code>\ifglssacrshortcuts</code>	15	<code>\KV@glslink@hyperfalse</code>	52, 65, 70
<code>\ifglssautomake</code>	98, 110, 112	<code>\KV@glslink@noindexfalse</code>	65, 66
<code>\ifglssentrycounter</code>	25	<code>\KV@glslink@noindextrue</code>	66, 70
<code>\ifglssentryexists</code>	30, 31, 40, 41, 43, 44, 51, 129, 130, 147, 148	L	
<code>\ifglssfieldeq</code>	128	<code>\LaTeX</code>	108, 109
<code>\ifglshasfield</code>	22, 66, 236	<code>\leaders</code>	281
<code>\ifglshaslong</code>	85, 86	<code>\leavevmode</code>	27, 49
		<code>\let</code>	5, 7, 8, 10, 11, 13–16, 18–20, 22, 26, 27, 37, 38, 42, 44,

<code>\NoCaseChange</code>	76–78, 264–272	<code>nopostdot</code>	
<code>\noexpand</code>	9, 17, 33, 35, 36, 92, 93, 106, 107, 111, 114, 141, 142, 155, 280	false	13
<code>\nofiles</code>	109	numbers	15
<code>\noindent</code>	109, 110	record	7, 10, 47, 110, 307
<code>\nopostdesc</code>	27, 40, 41, 99, 132	shortcuts	15
<code>\nr</code>	7, 10, 12, 15–17, 49, 99	ac	15
<code>\ns@GLSxtrfull</code>	159	all	15
<code>\ns@Glsxtrfull</code>	158	false	15
<code>\ns@glxtrfull</code>	157	none	15
<code>\ns@GLSxtrfullpl</code>	160	true	15
<code>\ns@Glsxtrfullpl</code>	159	style	18
<code>\ns@glxtrfullpl</code>	159	stylemods	18
<code>\ns@GLSxtrlong</code>	163	symbols	15, 132
<code>\ns@Glsxtrlong</code>	163	undefaction	30, 31
<code>\ns@glxtrlong</code>	162	error	6
<code>\ns@GLSxtrlongpl</code>	166	warn	6
<code>\ns@Glsxtrlongpl</code>	166	xindy	33, 34
<code>\ns@glxtrlongpl</code>	165	<code>\PackageError</code>	6, 9, 17, 19, 38, 41, 42, 63, 64, 66, 74, 75, 80–82, 88, 90, 92, 94, 95, 97, 103, 113, 115, 170–173, 280
<code>\ns@GLSxtrshort</code>	162	<code>\PackageWarning</code>	13
<code>\ns@Glsxtrshort</code>	161	<code>\PackageWarningNoLine</code>	13
<code>\ns@glxtrshort</code>	160, 161	<code>\pageref</code>	25
<code>\ns@GLSxtrshortpl</code>	165	<code>\par</code>	109, 110, 286, 287, 296
<code>\ns@Glsxtrshortpl</code>	164	<code>\parindent</code>	287, 296
<code>\ns@glxtrshortpl</code>	164	<code>\PassOptionsToPackage</code>	5
<code>\null</code>	17	<code>\preglossarypreamble</code>	25
<code>\number</code>	87–90, 111	<code>\preto</code>	66
<code>\numexpr</code>	87, 88, 90	<code>\print@noop@unsrtglossaryunit</code>	9, 10
O		<code>\print@op@unsrtglossaryunit</code>	11
<code>\or</code>	7, 10, 11, 16, 38, 49	<code>\printabbreviations</code>	13
<code>\org@glossaryentrynumbers</code>	43, 99	<code>\printglossaries</code>	96, 108
<code>\org@glossarytitle</code>	98, 99	<code>\printglossary</code>	13, 96, 108
<code>\org@ifKV@glslink@hyper</code>	49, 50	<code>\printglossary options</code>	
P		nonumberlist	46
<code>\p@gl@hyp@opt</code>	68	type	98
package options:		<code>\printnoidxglossaries</code>	108
abbreviations	13	<code>\printnoidxglossary</code>	96, 108
accsupp	17, 117	<code>\printnumbers</code>	15, 133
acronym	13	<code>\printsymbols</code>	15, 132
automake	98, 108, 112	<code>\printunsrtglossary</code>	114
true	113	<code>\printunsrtglossaryhandler</code>	115
autoseeindex	19	<code>\printunsrtglossaryunit</code>	10, 11, 115
false	18	<code>\printunsrtglossaryunitsetup</code>	115
docdef	11, 37, 38, 80, 88	<code>\ProcessOptions</code>	280
false	38	<code>\ProcessOptionsX</code>	18
restricted	12	<code>\protect</code>	76–78, 125–128, 153, 156, 173–183, 185–200, 202–235, 237– 240, 242–244, 246–251, 253–261, 264–272
true	38		
nonumberlist	43		

<code>\protected@csedef</code>	25, 287	<code>\settowidth</code>	94, 287–295
<code>\protected@csxdef</code>	25, 287	<code>\setupglossaries</code>	5, 19
<code>\protected@edef</code>		<code>\sfcode</code>	148, 286
.....	42, 92, 102, 103, 116, 140, 141, 155	<code>\space</code>	6, 9, 33, 39, 41, 66,
<code>\protected@write</code>	8, 9, 45, 95, 96, 110, 112, 113		80–82, 88, 90–92, 94–97, 99, 107, 109,
<code>\protected@xdef</code>	116		113, 115, 148, 152, 156, 174, 286, 287, 295
<code>\providecommand</code>	13, 21,	<code>\spacefactor</code>	148, 154, 286
	33, 45, 64, 66, 82, 91, 95, 106, 107, 112, 280	<code>\string</code> ...	6, 8, 9, 33, 34, 38, 39, 41, 42, 45,
<code>\ProvidesFile</code>	279		63, 64, 66, 74, 75, 80–82, 88, 90–92, 94–
<code>\ProvidesPackage</code>	5, 280		99, 106–110, 112, 113, 115, 136–139, 141
		<code>\strut</code>	281–286
		<code>\subglossentry</code>	99, 116, 281–286, 296
Q			
<code>\quotechar</code>	143	T	
R			
<code>\raggedright</code>	283, 285	<code>\tablehead</code>	284, 285
<code>\relax</code> ...	7, 8, 10–12, 14–17, 19, 22, 38, 42,	<code>\tabletail</code>	284, 285
	43, 45, 48, 49, 68, 74, 82, 90, 91, 95, 96,	<code>\tabularnewline</code>	281–286
	98, 99, 101, 102, 104, 110–113, 116, 141,	<code>\TeX</code>	108
	142, 144, 146, 148, 152–154, 245, 288–297	<code>\texorpdfstring</code>	22, 75–78, 273–278
<code>relsize</code> package	202	<code>textcase</code> package	262
<code>\renewcommand</code>	6, 10–13, 16–19, 26–31, 37–	<code>\textsc</code>	188
	39, 41–48, 63, 65, 67, 69, 70, 74, 79–82,	<code>\textsmaller</code>	202
	85, 86, 88–99, 102–104, 106, 107, 115,	<code>\texttt</code>	107–110
	132, 134–139, 144–148, 156, 157, 171–	<code>\the</code>	92, 93, 105, 111, 144, 155, 173–181, 183,
	235, 237–244, 246–262, 281–286, 296, 297		185, 187–195, 198, 200, 202–208, 211–
<code>\renewenvironment</code>	281–285, 295		225, 227–230, 232–234, 237–248, 250–261
<code>\renewglossarystyle</code>	281–285, 295	<code>\theglentrycounter</code>	8, 51
<code>\renewrobustcmd</code>	50, 70	<code>\theHglentrycounter</code>	8, 51
<code>\RequireGlossariesExtraLang</code>	279	<code>\theindex</code>	140
<code>\RequirePackage</code>	5, 17, 18, 280	<code>\this@dialect</code>	279
<code>\reserved@a</code>	150	<code>\toks@</code>	105, 144
<code>\reserved@b</code>	150	<code>tracklang</code> package	112
<code>\reserved@d</code>	150	U	
<code>\RestoreAcronyms</code>	94	<code>\undef</code>	10, 11, 145
<code>\romannumeral</code>	287, 288, 295	<code>\underline</code>	146
S			
<code>\s@glshyp@opt</code>	68	<code>\unskip</code>	27, 37, 281
<code>\s@glstr@enabletagging</code>	145	<code>\usepackage</code>	109, 110
<code>\s@printunsrtglossary</code>	113, 115	V	
<code>\seealso</code>	33, 34	<code>\val</code>	7, 10, 12, 15–17, 49, 99
<code>\seename</code>	32	W	
<code>\setabbreviationstyle</code>	94, 174, 183	<code>\warn@nomakeglossaries</code>	96
<code>\setacronymstyle</code>	93, 94	<code>\warn@noprintglossary</code>	96, 99
<code>\setentrycounter</code>	104	<code>\write</code>	33, 82, 90, 91, 95, 106, 107, 112
<code>\SetGenericNewAcronym</code>	94	X	
<code>\setglossarystyle</code>	18, 99, 281, 297	<code>\x</code>	105
<code>\setkeys</code>	7, 18, 19, 50, 67, 92, 99, 153, 154	<code>\xcapitalisewords</code>	134
<code>\setlength</code>	43, 287, 296		

\xdef	99, 115	xkeyval package	5
\xifinlist	86	\XKV@checkchoice	46
\xifinlistcs	23	\XKV@plfalse	46
xindy	298	\XKV@resa	46
xindy	95	\XKV@sttrue	46