

glossaries-extra.sty v1.20: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2017-09-11

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	21
1.3 Modifications to Commands Provided by glossaries	27
1.3.1 Existence Checks	31
1.3.2 Document Definitions	39
1.3.3 Existing Glossary Style Modifications	43
1.3.4 Entry Formatting, Hyperlinks and Indexing	47
1.3.5 Entry Counting	81
1.3.6 Acronym Modifications	94
1.3.7 Indexing and Displaying Glossaries	96
1.4 Integration with glossaries-accsupp	119
1.5 Categories	130
1.6 Abbreviations	153
1.6.1 Abbreviation Styles Setup	172
1.6.2 Predefined Styles (Default Font)	175
1.6.3 Predefined Styles (Small Capitals)	190
1.6.4 Predefined Styles (Fake Small Capitals)	204
1.6.5 Predefined Styles (Emphasized)	217
1.6.6 Predefined Styles (User Parentheses Hook)	238
1.6.7 Predefined Styles (Hyphen)	247
1.6.8 Predefined Styles (No Short on First Use)	261
1.7 Using Entries in Headings	264
1.8 Multi-Lingual Support	281
2 Style Adjustments (glossaries-extra-stylemods.sty)	283
2.1 Package Initialisation	283
2.2 List-Like Styles	284
2.3 Longtable Styles	284
2.4 Long Ragged Styles	285
2.5 Supertabular Styles	287
2.6 Super Ragged Styles	288
2.7 Inline Style	289
2.8 Tree Styles	289
Glossary	301
Change History	302

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2017/09/11 v1.20 (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{\glsxtr@dooption}[1]{\setupglossaries{#1}}%
8 \let\@glsxtr@declareoption\@gls@declareoption
9 }
10 {%
```

Not already loaded, so pass options to glossaries.

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

Set the defaults.

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

Declare package options.

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

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

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

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

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

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

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

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

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

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

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

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

```
41 \newcommand*\@glxtr@warn@onexistsordo}[1]{%
42 \GlossariesExtraWarning{\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*\@glxtr@redef@for@gl@sentries}{}
```

`\f@for@gl@sentries`

```
48 \newcommand*\@glxtr@do@redef@for@gl@sentries}{%
49 \renewcommand*\f@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\glxtrundefaction\@glxtr@warn@undefaction
68     \let\glxtr@warnonexistsordo\@glxtr@warn@onexistsordo
69     \let\@glxtr@redef@forglsentries\@glxtr@do@redef@forglsentries
70   \or
71     \let\glxtrundefaction\@glxtr@err@undefaction
72     \let\glxtr@warnonexistsordo\@gobble
73     \let\@glxtr@redef@forglsentries\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.

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

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

`\glxtr@recordsee` Does nothing by default.

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

`\ultnumberformat`

```
78 \newcommand*\@glxtr@defaultnumberformat}{glsnumberformat}%

```

`\ultNumberFormat`

```
79 \newcommand*\GlsXtrSetDefaultNumberFormat}[1]{%
80 \renewcommand*\@glxtr@defaultnumberformat}{#1}%
81 }%

```

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

```

82 \newcommand*\@@glxtr@record}[3]{%
83   \begingroup
84     \let\@glsnumberformat\@glxtr@defaultnumberformat
85     \def\@glxtr@thevalue{}%
86     \def\@glxtr@theHvalue{\@glxtr@thevalue}%
87     \let\@glxtr@org@theHvalue\@glxtr@theHvalue
88     \ifcsdef{glo@#2@counter}%

```

```

89  {%
90  \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
91  }%
92  {%

```

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

```

93  \def\@gls@counter{page}%
94  }%
95  \setkeys{#3}{#1}%
96  \ifKV@glslink@noindex
97  \else
98  \glswriteentry{#2}%
99  {%

```

Check if the value has been set.

```

100  \ifdefempty{\@glsxtr@thevalue}%
101  {%

```

Key the value hasn't been set, but check if the H value has been set. (Not particularly likely, but allow for it.)

```

102  \ifx\@glsxtr@org@theHvalue\@glsxtr@theHvalue
103  \else
104  \let\theHglentrycounter\@glsxtr@theHvalue
105  \fi

```

Save the entry counter.

```

106  \glsxtr@saveentrycounter

```

Temporarily redefine \@@do@@wrglossary for use with \glsxtr@@do@@wrglossary.

```

107  \let\@@do@@wrglossary\@glsxtr@dorecord
108  }%
109  {%

```

the value has been set, so there's no need to defer writing the location value. (If it's dependent on the page counter, the counter key should be set instead.)

```

110  \let\theglentrycounter\@glsxtr@thevalue
111  \let\theHglentrycounter\@glsxtr@theHvalue
112  \let\@@do@@wrglossary\@glsxtr@dorecordnodefer
113  }%
114  \ifx\@glsxtr@record@setting\@glsxtr@record@setting@alsoindex
115  \glsxtr@@do@@wrglossary{#2}%
116  \else

```

No need to escape special characters, but need to save the label.

```

117  \edef\@gls@label{\glsdetoklabel{#2}}%
118  \@@do@@wrglossary
119  \fi
120  }%
121  \fi
122 \endgroup
123 }

```


glsxtr@dorecord If record=alsoindex is used, then \@glslocref may have been escaped, but this isn't appropriate here.

```
124 \newcommand*\@glsxtr@dorecord{%
125   \global\let\@glsrecordlocref\theglsentrycounter
126   \let\@glsxtr@orgprefix\@glo@counterprefix
127   \ifx\theglsentrycounter\theHglentrycounter
128     \def\@glo@counterprefix{%
129       \else
130         \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
131           {\theglsentrycounter}{\theHglentrycounter}}%
132       }%
133       \@do@gls@getcounterprefix
134     \fi
135     \protected@write\@auxout{\let\@glsrecordlocref\relax}{\string\glsxtr@record
136       {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
137       {\@glsrecordlocref}}%
138     \@glsxtr@counterrecordhook
139     \let\@glo@counterprefix\@glsxtr@orgprefix
140 }
```

dorecordnodefer As above, but don't defer expansion of location. This uses \theglsentrycounter directly for the location rather than \@glslocref since there's no need to guard against premature expansion of the page counter.

```
141 \newcommand*\@glsxtr@dorecordnodefer{%
142   \ifx\theglsentrycounter\theHglentrycounter
143     \protected@write\@auxout{}{\string\glsxtr@record
144       {\@gls@label}{\@gls@counter}{\@glsnumberformat}%
145       {\theglsentrycounter}}%
146   \else
147     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
148       {\theglsentrycounter}{\theHglentrycounter}}%
149     }%
150     \@do@gls@getcounterprefix
151     \protected@write\@auxout{}{\string\glsxtr@record
152       {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
153       {\theglsentrycounter}}%
154   \fi
155   \@glsxtr@counterrecordhook
156 }
```

r@recordcounter

```
157 \newcommand*\@@glsxtr@recordcounter}{%
158   \@glsxtr@noop@recordcounter
159 }
```

p@recordcounter

```
160 \newcommand*\@glsxtr@noop@recordcounter}[1]{%
161   \PackageError{glossaries-extra}{\string\GlsXtrRecordCounter\space
```

```

162 requires record=only or record=alsoindex package option}{}%
163 }

```

p@recordcounter

```

164 \newcommand*{\@glsxtr@op@recordcounter}[1]{%
165 \eappto\@glsxtr@counterrecordhook{\noexpand\@glsxtr@docounterrecord{#1}}%
166 }

```

lsxtr@recordsee Deal with \glssee in record mode.

```

167 \newcommand*{\@glsxtr@recordsee}[2]{%
168 \def\@gls@xref{#2}%
169 \@onelevel@sanitize\@gls@xref
170 \protected@write\@auxout{}{\string\glsxtr@recordsee{#1}{\@gls@xref}}%
171 }

```

srtglossaryunit

```

172 \newcommand{\printunsrtglossaryunit}{%
173 \print@noop@unsrtglossaryunit
174 }

```

tr@setup@record Initialise.

```

175 \newcommand*{\glsxtr@setup@record}{}

```

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

```

176 \newcommand*{\glsxtr@indexonly@saveentrycounter}{%
177 \ifKV@glslink@noindex
178 \else
179 \glsxtr@saveentrycounter
180 \fi
181 }

```

addloclistfield

```

182 \newcommand*{\glsxtr@addloclistfield}{%
183 \key@ifundefined{glossentry}{loclist}%
184 {%
185 \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
186 \appto\@gls@keymap{,}{loclist}{loclist}}%
187 \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
188 \appto\@newglossaryentryposthook{%
189 \gls@assign@field}{\@glo@label}{loclist}{\@glo@loclist}%
190 }%
191 \glssetnoexpandfield{loclist}%
192 }%
193 }%

```

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

```

194 \key@ifundefined{glossentry}{location}%
195 {%
196 \define@key{glossentry}{location}{\def\@glo@location{##1}}%

```

```

197 \appto\@gls@keymap{,{location}{location}}%
198 \appto\@newglossaryentryprehook{\def\@glo@location{}}%
199 \appto\@newglossaryentryposthook{%
200 \gls@assign@field{\@glo@label}{location}{\@glo@location}%
201 }%
202 \glssetnoexpandfield{location}%
203 }%
204 {}%

```

Add a key to store the group heading.

```

205 \key@ifundefined{glossentry}{group}%
206 {%
207 \define@key{glossentry}{group}{\def\@glo@group{##1}}%
208 \appto\@gls@keymap{,{group}{group}}%
209 \appto\@newglossaryentryprehook{\def\@glo@group{}}%
210 \appto\@newglossaryentryposthook{%
211 \gls@assign@field{\@glo@label}{group}{\@glo@group}%
212 }%
213 \glssetnoexpandfield{group}%
214 }%
215 {}%
216 }

```

`@record@setting` Keep track of the record package option.

```

217 \newcommand*\@glsxtr@record@setting{off}

```

`ttting@alsoindex`

```

218 \newcommand*\@glsxtr@record@setting@alsoindex{alsoindex}

```

Now define the record package option.

```

219 \define@choicekey{glossaries-extra.sty}{record}[\val\nr]%
220 {off,only,alsoindex}%
221 [only]%
222 {%
223 \let\@glsxtr@record@setting\val
224 \ifcase\nr\relax

```

Don't record.

```

225 \def\glsxtr@setup@record{%
226 \renewcommand*\@do@seeglossary{\@glsxtr@org@doseeglossary}%
227 \renewcommand*\@glsxtr@record}[3]{}%
228 \let\@do@wrglossary\glsxtr@do@wrglossary
229 \let\@gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
230 \let\glsxtrundefaction\@glsxtr@err@undefaction
231 \let\glsxtr@warnonexistsordo\@gobble
232 \let\@glsxtr@recordcounter\@glsxtr@noop@recordcounter
233 \def\printunsrtglossaryunit{\print@noop@unsrtglossaryunit}%
234 \undef\glsxtrsetaliasnoindex
235 }%
236 \or

```

Only record (don't index).

```
237 \def\glxtr@setup@record{%
238   \glxtr@autoseeindexfalse
239   \let\@do@seeglossary\glxtr@recordsee
240   \let\glxtr@record\@glxtr@record
241   \let\@do@wrglossary\gobble
242   \let\gls@saveentrycounter\relax
243   \let\glxtrundefaction\glxtr@warn@undefaction
244   \let\glxtr@warnonexistsordo\glxtr@warn@onexistsordo
245   \glxtr@addloclistfield
246   \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
247   \let\@glxtr@recordcounter\glxtr@op@recordcounter
248   \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
```

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

```
249 \def\glxtrsetaliasnoindex{}%
```

\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.

```
250 \ifdef\gls@setupsort@none{\gls@setupsort@none}{}%
251 }%
252 \or
```

Record and index.

```
253 \def\glxtr@setup@record{%
254   \renewcommand*{\@do@seeglossary}{\@glxtr@org@doseeglossary}%
255   \let\glxtr@record\@glxtr@record
256   \let\@do@wrglossary\glxtr@do@wrglossary
257   \let\gls@saveentrycounter\glxtr@indexonly@saveentrycounter
258   \let\glxtrundefaction\glxtr@warn@undefaction
259   \let\glxtr@warnonexistsordo\glxtr@warn@onexistsordo
260   \glxtr@addloclistfield
261   \let\@glxtr@recordcounter\glxtr@op@recordcounter
262   \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
263   \undef\glxtrsetaliasnoindex
264 }%
265 \fi
266 }
```

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

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

```
267 \newcount\@glxtr@docdefval
```

Need to provide conditional commands that are backward compatible:

\if@glxtrdocdef

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

lsxtrdocdeftrue

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

lsxtrdocdeffalse

```
270 \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.

```
271 \define@choicekey{glossaries-extra.sty}{docdef}{\val\nr}%  
272 {false,true,restricted}[true]%  
273 {%  
274   \@glxtr@docdefval=\nr\relax  
275   \ifnum\@glxtr@docdefval=2\relax  
276     \renewcommand*{\@glsdofexistsorwarn}{\glsdofexists}%  
277   \fi  
278 }
```

docdefrestricted

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

dofexistsorwarn

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).

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

indexcrossrefs

Automatically index cross references at the end of the document

```
281 \define@boolkey{glossaries-extra.sty}{@glxtr}{indexcrossrefs}[true]{%  
282   \if@glxtrindexcrossrefs  
283   \else  
284     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%  
285   \fi  
286 }
```

Switch off since this can increase the build time.

```
287 \@glxtrindexcrossrefsfalse
```

But allow see key to switch it on automatically.

oindexcrossrefs

```
288 \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.

```
289 \define@boolkey{glossaries-extra.sty}{@glxtr@}{autoseeindex}[true]{%  
290 }  
291 \@glxtr@autoseeindextrue
```

```

iesExtraWarning  Allow users to suppress warnings.
292 \newcommand*{\GlossariesExtraWarning}[1]{\PackageWarning{glossaries-extra}{#1}}

raWarningNoLine  Allow users to suppress warnings.
293 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
294 \PackageWarningNoLine{glossaries-extra}{#1}}

295 \@glsxtr@declareoption{nowarn}{%
296 \let\GlossariesExtraWarning\@gobble
297 \let\GlossariesExtraWarningNoLine\@gobble
298 \glsxtr@doooption{nowarn}%
299 }

postdot  Shortcut for nopostdot=false
300 \@glsxtr@declareoption{postdot}{%
301 \glsxtr@doooption{nopostdot=false}%
302 }

glsxtrabbrvtype  Glossary type for abbreviations.
303 \newcommand*{\glsxtrabbrvtype}{\glsdefaulttype}

abbreviationsdef  Set by abbreviations option.
304 \newcommand*{\@glsxtr@abbreviationsdef}{}

abbreviationsdef
305 \newcommand*{\@glsxtr@doabbreviationsdef}{%
306 \ifpackageloaded{babel}%
307 {\providecommand{\abbreviationsname}{\acronymname}}%
308 {\providecommand{\abbreviationsname}{Abbreviations}}%
309 \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
310 \renewcommand*{\glsxtrabbrvtype}{abbreviations}%
311 \newcommand*{\printabbreviations}[1][ ]{%
312 \printglossary[type=\glsxtrabbrvtype,##1]%
313 }%
314 \disable@keys{glossaries-extra.sty}{abbreviations}%

    If the acronym option hasn't been used, change \acronymtype to \glsxtrabbrvtype.
315 \ifglsacronym
316 \else
317 \renewcommand*{\acronymtype}{\glsxtrabbrvtype}%
318 \fi
319 }%

abbreviations  If abbreviations, create a new glossary type for abbreviations.
320 \@glsxtr@declareoption{abbreviations}{%
321 \let\@glsxtr@abbreviationsdef\@glsxtr@doabbreviationsdef
322 }

```

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

```
323 \newcommand*\GlsXtrDefineAbbreviationShortcuts}{%
324   \newcommand*\ab{\cGls}%
325   \newcommand*\abp{\cGlspl}%
326   \newcommand*\as{\glxtrshort}%
327   \newcommand*\asp{\glxtrshortpl}%
328   \newcommand*\al{\glxtrlong}%
329   \newcommand*\alp{\glxtrlongpl}%
330   \newcommand*\af{\glxtrfull}%
331   \newcommand*\afp{\glxtrfullpl}%
332   \newcommand*\Ab{\cGls}%
333   \newcommand*\Abp{\cGlspl}%
334   \newcommand*\As{\Glsxtrshort}%
335   \newcommand*\Asp{\Glsxtrshortpl}%
336   \newcommand*\Al{\Glsxtrlong}%
337   \newcommand*\Alp{\Glsxtrlongpl}%
338   \newcommand*\Af{\Glsxtrfull}%
339   \newcommand*\Afp{\Glsxtrfullpl}%
340   \newcommand*\AB{\cGLS}%
341   \newcommand*\ABP{\cGLSpl}%
342   \newcommand*\AS{\GLSxtrshort}%
343   \newcommand*\ASP{\GLSxtrshortpl}%
344   \newcommand*\AL{\GLSxtrlong}%
345   \newcommand*\ALP{\GLSxtrlongpl}%
346   \newcommand*\AF{\GLSxtrfull}%
347   \newcommand*\AFP{\GLSxtrfullpl}%
348   \newcommand*\newabbr{\newabbreviation}%
```

Disable this command after it's been used.

```
349 \let\GlsXtrDefineAbbreviationShortcuts\relax
350 }
```

fineAcShortcuts Enable shortcut commands for the abbreviations, but uses the analogous commands provided by glossaries.

```
351 \newcommand*\GlsXtrDefineAcShortcuts}{%
352   \newcommand*\ac{\cGls}%
353   \newcommand*\acp{\cGlspl}%
354   \newcommand*\acs{\glxtrshort}%
355   \newcommand*\acsp{\glxtrshortpl}%
356   \newcommand*\acl{\glxtrlong}%
357   \newcommand*\aclp{\glxtrlongpl}%
358   \newcommand*\acf{\glxtrfull}%
359   \newcommand*\acfp{\glxtrfullpl}%
360   \newcommand*\Ac{\cGls}%
361   \newcommand*\Acp{\cGlspl}%
362   \newcommand*\Acs{\Glsxtrshort}%
363   \newcommand*\Acsp{\Glsxtrshortpl}%
364   \newcommand*\Acl{\Glsxtrlong}%
```

```

365 \newcommand*\Ac1p}{\Glsxtrlongpl}%
366 \newcommand*\Acf}{\Glsxtrfull}%
367 \newcommand*\Acfp}{\Glsxtrfullpl}%
368 \newcommand*\AC}{\cGLS}%
369 \newcommand*\ACP}{\cGLSp1}%
370 \newcommand*\ACS}{\GLSxtrshort}%
371 \newcommand*\ACSP}{\GLSxtrshortpl}%
372 \newcommand*\ACL}{\GLSxtrlong}%
373 \newcommand*\ACLP}{\GLSxtrlongpl}%
374 \newcommand*\ACF}{\GLSxtrfull}%
375 \newcommand*\ACFP}{\GLSxtrfullpl}%
376 \newcommand*\newabbr}{\newabbreviation}%

```

Disable this command after it's been used.

```

377 \let\GlsXtrDefineAcShortcuts\relax
378 }

```

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

```

379 \newcommand*\GlsXtrDefineOtherShortcuts}{%
380 \newcommand*\newentry}{\newglossaryentry}%
381 \ifdef\printsymbols
382 {%
383 \newcommand*\newsym}{\glsxtrnewsymbol}%
384 }{}%
385 \ifdef\printnumbers
386 {%
387 \newcommand*\newnum}{\glsxtrnewnumber}%
388 }{}%
389 \let\GlsXtrDefineOtherShortcuts\relax
390 }

```

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.

```

391 \newcommand*\@glsxtr@setupshortcuts}{%

```

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

```

392 \newcommand*\@glsxtr@shortcutsval}{\ifglsacrshortcuts 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).

```

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

```



```

395 \let\@glxtr@shortcutsval\val
396 \ifcase\nr\relax % acronyms
397   \renewcommand*\@glxtr@setupshortcuts}{%
398     \glsacrshortcutstrue
399     \DefineAcronymSynonyms
400   }%
401 \or % acro
402   \renewcommand*\@glxtr@setupshortcuts}{%
403     \glsacrshortcutstrue
404     \DefineAcronymSynonyms
405   }%
406 \or % abbreviations
407   \renewcommand*\@glxtr@setupshortcuts}{%
408     \GlsXtrDefineAbbreviationShortcuts
409   }%
410 \or % abbr
411   \renewcommand*\@glxtr@setupshortcuts}{%
412     \GlsXtrDefineAbbreviationShortcuts
413   }%
414 \or % other
415   \renewcommand*\@glxtr@setupshortcuts}{%
416     \GlsXtrDefineOtherShortcuts
417   }%
418 \or % all
419   \renewcommand*\@glxtr@setupshortcuts}{%
420     \glsacrshortcutstrue
421     \DefineAcronymSynonyms
422     \GlsXtrDefineAbbreviationShortcuts
423     \GlsXtrDefineOtherShortcuts
424   }%
425 \or % true
426   \renewcommand*\@glxtr@setupshortcuts}{%
427     \glsacrshortcutstrue
428     \DefineAcronymSynonyms
429     \GlsXtrDefineAbbreviationShortcuts
430     \GlsXtrDefineOtherShortcuts
431   }%
432 \or % none, false
433   \renewcommand*\@glxtr@setupshortcuts}{}%
434 \or % ac
435   \renewcommand*\@glxtr@setupshortcuts}{%
436     \glsacrshortcutstrue
437     \GlsXtrDefineAcShortcuts
438   }%
439 \fi
440 }

```

lsxtr@doaccsupp

```
441 \newcommand*\@glxtr@doaccsupp}{}
```

accsupp If accsupp, load glossaries-accsupp package.

```
442 \@glsxtr@declareoption{accsupp}{%
443 \renewcommand*{\@glsxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}
```

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

```
444 \newcommand{\glsxtrNoGlossaryWarning}[1]{%
445 \@glsxtr@defaultnoglossarywarning{#1}%
446 }
```

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

```
447 \define@choicekey{glossaries-extra.sty}{nomissingglstext}[\val\nr]%
448 {true,false}[true]{%
449 \ifcase\nr\relax % true
450 \renewcommand{\glsxtrNoGlossaryWarning}[1]{%
451 \null
452 }%
453 \else % false
454 \renewcommand{\glsxtrNoGlossaryWarning}[1]{%
455 \@glsxtr@defaultnoglossarywarning{#1}%
456 }%
457 \fi
458 }
```

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

xtr@redefstyles

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

stylemods

```
460 \define@key{glossaries-extra.sty}{stylemods}[default]{%
461 \ifstrequal{#1}{default}%
462 {%
463 \renewcommand*{\@glsxtr@redefstyles}{%
464 \RequirePackage{glossaries-extra-stylemods}}%
465 }%
466 {%
467 \renewcommand*{\@glsxtr@redefstyles}{}%
468 \@for\@glsxtr@tmp:=#1\do{%
469 \IfFileExists{glossary-\@glsxtr@tmp.sty}%
470 {%
471 \eappto\@glsxtr@redefstyles{%
472 \noexpand\RequirePackage{glossary-\@glsxtr@tmp}}%
473 }%
474 {%
475 \PackageError{glossaries-extra}%
476 {Glossaries style package ‘glossary-\@glsxtr@tmp.sty’
477 doesn’t exist (did you mean to use the ‘style’ key?)}%
478 {The list of values (#1) in the ‘stylemods’ key should
```

```

479         match the glossary-xxx.sty files provided with
480         glossaries.sty}%
481     }%
482 }%
483 \appto\@glsxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
484 }%
485 }

```

glsxtr@do@style

```
486 \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.

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

Defer actual style change:

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

Set this as the default style:

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

Set this style:

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

```
491 }%
```

```
492 }
```

Pass all other options to glossaries.

```
493 \DeclareOptionX*{%
```

```
494 \expandafter\glsxtr@dooption\expandafter{\CurrentOption}}
```

Process options.

```
495 \ProcessOptionsX
```

Load glossaries if not already loaded.

```
496 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
497 \@glsxtr@doaccsupp
```

g@doseeglossary Save original definition of \@do@seeglossary

```
498 \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.)

```
499 \let\@glsxtr@org@gloautosee\@glo@autosee
```

Check if user tried autoseeindex=false when it can't be supported.

```
500 \if@glsxtr@autoseeindex
```

```
501 \else
```

```

502 \ifdef\@glxtr@org@gloautosee
503 {}%
504 {\PackageError{glossaries-extra}{‘autoseeindex=false’ package
505 option requires at least v4.30 of glossaries.sty}%
506 {You need to update the glossaries.sty package}%
507 }
508 \fi

```

`\@glo@autosee` If `\@glo@autosee` has been defined (glossaries v4.30 onwards), redefine it to test the `autoseeindex` option.

```

509 \ifdef\@glo@autosee
510 {%
511 \renewcommand*{\@glo@autosee}{%
512 \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
513 }%
514 {}

```

`checkseeallowed` Don't prohibit the use of the `see` key before the indexing files have been opened if the automatic `see` indexing has been disabled, since it's no longer an issue.

```

515 \renewcommand*{\gls@checkseeallowed}{%
516 \if@glxtr@autoseeindex\@gls@see@noindex\fi
517 }

```

Define abbreviations glossaries if required.

```

518 \@glxtr@abbreviationsdef
519 \let\@glxtr@abbreviationsdef\relax

```

Setup shortcuts if required.

```

520 \@glxtr@setupshortcuts

```

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

```

521 \@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`:

```

522 \renewcommand{\glxtr@doooption}[1]{\setupglossaries{#1}}%

```

Now define the user command:

```

523 \newcommand*{\glossariesextrasetup}[1]{%
524 \let\glxtr@setup@record\relax
525 \let\@glxtr@setupshortcuts\relax
526 \let\@glxtr@redef@for@gl@sentries\relax
527 \setkeys{glossaries-extra.sty}{#1}%
528 \@glxtr@abbreviationsdef
529 \let\@glxtr@abbreviationsdef\relax
530 \@glxtr@setupshortcuts
531 \glxtr@setup@record
532 \@glxtr@redef@for@gl@sentries
533 }

```

`@do@wrglossary` Save original definition of `\@do@wrglossary`.
534 `\let\glxtr@do@wrglossary\@do@wrglossary`

`saveentrycounter` Save original definition of `\@gls@saveentrycounter`.
535 `\let\glxtr@saveentrycounter\@gls@saveentrycounter`

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

Set up record option if required.
537 `\glxtr@setup@record`

Disable preamble-only options and switch on the undefined tag at the start of the document.
538 `\AtBeginDocument{%`
539 `\disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%`
540 `\def\@glxtrundeftag{\glxtrundeftag}%`
541 `}`

1.2 Extra Utilities

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

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

```
542 \newcommand{\glxtrifemptyglossary}[3]{%
543   \ifcsdef{glolist@#1}%
544   {%
545     \ifcsstring{glolist@#1}{,}{#2}{#3}%
546   }%
547   {%
548     \glxtrundefaction{Glossary type '#1' doesn't exist}{}%
549     #2%
550   }%
551 }
```

`trifkeydefined` Tests if the key given in the first argument has been defined.

```
552 \newcommand*\glxtrifkeydefined[3]{%
553   \key@ifundefined{glossentry}{#1}{#3}{#2}%
554 }
```

ovidestoragekey Like \glsaddstoragekey but does nothing if the key has already been defined.

```
555 \newcommand*\glsxtrprovidestoragekey}{%
556 \@ifstar\sglsxtr@provide@storagekey\@glsxtr@provide@storagekey
557 }
```

vide@storagekey Unstarred version.

```
558 \newcommand*\@glsxtr@provide@storagekey}[3]{%
559 \key@ifundefined{glossentry}{#1}%
560 {%
561 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
562 \appto\@gls@keymap{, #1}{#1}}%
563 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
564 \appto\@newglossaryentryposthook{%
565 \letcs{\@glo@tmp}{@glo@#1}%
566 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
567 }
```

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

\glsxtrusefield

```
568 \ifblank{#3}
569 {}%
570 {%
571 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
572 }%
573 }%
574 {%
```

Provide the no-link command if not already defined.

```
575 \ifblank{#3}
576 {}%
577 {%
578 \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
579 }%
580 }%
581 }
```

vide@storagekey Starred version.

```
582 \newcommand*\s@glsxtr@provide@storagekey}[1]{%
583 \key@ifundefined{glossentry}{#1}%
584 {%
585 \expandafter\newcommand\expandafter*\expandafter
586 {\csname gls@assign@#1@field\endcsname}[2]{%
587 \@gls@expand@field{##1}{#1}{##2}%
588 }%
589 }%
590 {}%
591 \@glsxtr@provide@addstoragekey{#1}%
592 }
```

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

`\GlsXtrFmtField`

```
593 \newcommand{\GlsXtrFmtField}{useri}
```

`tDefaultOptions`

```
594 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}
```

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

```
595 \newrobustcmd*{\glsxtrfmt}[3][ ]{%
596 \glsdoifexistsordo{#2}%
597 {%
598 \ifglshasfield{\GlsXtrFmtField}{#2}%
599 {%
600 \let\do@gls@link@checkfirsthyper\relax
601 \expandafter\@gls@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
602 {\csuse{\glscurrentfieldvalue}{#3}}%
603 }%
604 {#3}%
605 }%
606 {#3}%
607 }
```

`\glsxtrentryfmt` No link or indexing.

```
608 \ifdef\texorpdfstring
609 {
610 \newcommand*{\glsxtrentryfmt}[2]{%
611 \texorpdfstring{\@glsxtrentryfmt{#1}{#2}}{#2}%
612 }
613 }
614 {
615 \newcommand*{\glsxtrentryfmt}{\@glsxtrentryfmt}
616 }
```

`@glsxtrentryfmt`

```
617 \newrobustcmd*{\@glsxtrentryfmt}[2]{%
618 \glsdoifexistsordo
619 {%
620 \ifglshasfield{\GlsXtrFmtField}{#1}%
621 {%
622 \csuse{\glscurrentfieldvalue}{#2}%
623 }%
624 {#2}%
625 }%
626 {#2}%
627 }
```

`xtrfieldlistadd` 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.

```
628 \newcommand*\glxtrfieldlistadd}[3]{%
629   \listcsadd{glo@\glstdetoklabel{#1}@#2}{#3}%
630 }
```

`trfieldlistgadd` Similarly but uses `\listcsgadd`.

```
631 \newcommand*\glxtrfieldlistgadd}[3]{%
632   \listcsgadd{glo@\glstdetoklabel{#1}@#2}{#3}%
633 }
```

`trfieldlisteadd` Similarly but uses `\listcseadd`.

```
634 \newcommand*\glxtrfieldlisteadd}[3]{%
635   \listcseadd{glo@\glstdetoklabel{#1}@#2}{#3}%
636 }
```

`trfieldlistxadd` Similarly but uses `\listcsxadd`.

```
637 \newcommand*\glxtrfieldlistxadd}[3]{%
638   \listcsxadd{glo@\glstdetoklabel{#1}@#2}{#3}%
639 }
```

Now provide commands to iterate over these lists.

`fielddolistloop`

```
640 \newcommand*\glxtrfielddolistloop}[2]{%
641   \dolistcsloop{glo@\glstdetoklabel{#1}@#2}%
642 }
```

`fieldforlistloop`

```
643 \newcommand*\glxtrfieldforlistloop}[3]{%
644   \forlistcsloop{glo@\glstdetoklabel{#1}@#2}{#3}%
645 }
```

List element tests:

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

```
646 \newcommand*\glxtrfielddifyinlist}[5]{%
647   \ifyinlistcs{#3}{glo@\glstdetoklabel{#1}@#2}{#4}{#5}%
648 }
```

`rfieldxifyinlist` Expands item.

```
649 \newcommand*\glxtrfieldxifyinlist}[5]{%
650   \xifyinlistcs{#3}{glo@\glstdetoklabel{#1}@#2}{#4}{#5}%
651 }
```


`lstrifhasfield` A simpler alternative to `\ifglshasfield` that doesn't complain if the entry or the field doesn't exist. (No mapping is used.) Grouping is added to the unstarred version allow for nested use.

```
652 \newrobustcmd{\glstrifhasfield}{%
653   \@ifstar{\s@glstrifhasfield}{\@glstrifhasfield}%
654 }
```

`lstrifhasfield` Unstarred version adds grouping.

```
655 \newcommand{\@glstrifhasfield}[4]{%
656   {\s@glstrifhasfield{#1}{#2}{#3}{#4}}%
657 }
```

`lstrifhasfield` Starred version omits grouping.

```
658 \newcommand{\s@glstrifhasfield}[4]{%
659   \letcs\glscurrentfieldvalue{glo@\glsdetoklabel{#2}@#1}%
660   \ifdef\glscurrentfieldvalue
661     {%
662       \ifdefempty\glscurrentfieldvalue{#4}{#3}%
663     }%
664   {#4}%
665 }
```

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

```
666 \newcommand*{\glstrusefield}[2]{%
667   \@gls@entry@field{#1}{#2}%
668 }
```

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

```
669 \newcommand*{\Glsxtrusefield}[2]{%
670   \@gls@entry@field{#1}{#2}%
671 }
```

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

```
672 \newcommand*{\glstrdeffield}[2]{\csdef{glo@\glsdetoklabel{#1}@#2}}
```

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

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

`etfieldifexists`

```
674 \newcommand*{\glstrsetfieldifexists}[3]{\glsoifexists{#1}{#3}}
```

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

```
675 \newrobustcmd*\GlsXtrSetField}[3]{%
676   \glstrsetfieldifexists{#1}{#2}%
677   {\csdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
678 }
```

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

```

679 \newrobustcmd*{\GlsXtrLetField}[3]{%
680   \glsxtrsetfieldifexists{#1}{#2}%
681   {\cslet{glo@\glsdetoklabel{#1}@#2}{#3}}%
682 }

```

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

```

683 \newrobustcmd*{\csGlsXtrLetField}[3]{%
684   \glsxtrsetfieldifexists{#1}{#2}%
685   {\csletcs{glo@\glsdetoklabel{#1}@#2}{#3}}%
686 }

```

`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.

```

687 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
688   \glsxtrsetfieldifexists{#1}{#2}%
689   {\csletcs{glo@\glsdetoklabel{#1}@#2}{glo@\glsdetoklabel{#3}@#4}}%
690 }

```

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

```

691 \newrobustcmd*{\gGlsXtrSetField}[3]{%
692   \glsxtrsetfieldifexists{#1}{#2}%
693   {\csgdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
694 }

```

`xGlsXtrSetField`

```

695 \newrobustcmd*{\xGlsXtrSetField}[3]{%
696   \glsxtrsetfieldifexists{#1}{#2}%
697   {\protected@csxdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
698 }

```

`eGlsXtrSetField`

```

699 \newrobustcmd*{\eGlsXtrSetField}[3]{%
700   \glsxtrsetfieldifexists{#1}{#2}%
701   {\protected@csedef{glo@\glsdetoklabel{#1}@#2}{#3}}%
702 }

```

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

```

703 \ifglsentrycounter
704   \newcommand*{\glsxtrpageref}[1]{\pageref{glsentry-\glsdetoklabel{#1}}}
705 \else
706   \ifglsesubentrycounter
707     \newcommand*{\glsxtrpageref}[1]{\pageref{glsentry-\glsdetoklabel{#1}}}
708   \else
709     \newcommand*{\glsxtrpageref}[1]{\gls{#1}}
710   \fi
711 \fi

```

lossary preamble

```
712 \newcommand{\apptoglossarypreamble}[2][\glsdefaulttype]{%
713   \ifcsdef{glolist@#1}%
714   {%
715     \ifcsundef{@glossarypreamble@#1}%
716     {\csdef{@glossarypreamble@#1}{}}%
717     {}%
718     \csappto{@glossarypreamble@#1}{#2}%
719   }%
720   {%
721     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
722   }%
723 }
```

lossary preamble

```
724 \newcommand{\preglossarypreamble}[2][\glsdefaulttype]{%
725   \ifcsdef{glolist@#1}%
726   {%
727     \ifcsundef{@glossarypreamble@#1}%
728     {\csdef{@glossarypreamble@#1}{}}%
729     {}%
730     \cspretto{@glossarypreamble@#1}{#2}%
731   }%
732   {%
733     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
734   }%
735 }
```

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 `\glsxtrpostlongdescription` instead.

ewglossaryentry

```
736 \renewcommand*{\longnewglossaryentry}{%
737   \@ifstar\glsxtr@s@longnewglossaryentry\glsxtr@longnewglossaryentry
738 }
```

ewglossaryentry Starred version.

```
739 \newcommand{\@glsxtr@s@longnewglossaryentry}[3]{%
740   \glsdoifnoexists{#1}%
741   {%
742     \bgroup
743     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
```

```

744 \long\def\@newglossaryentryprehook{%
745 \long\def\@glo@desc{#3}%
746 \@org@newglossaryentryprehook
747 }%
748 \renewcommand*\gls@assign@desc}[1]{%
749 \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
750 \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
751 }
752 \gls@defglossaryentry{#1}{#2}%
753 \egroup
754 }%
755 }

```

`\newglossaryentry` Unstarred version.

```

756 \newcommand{\@glsxtr@longnewglossaryentry}[3]{%
757 \glsdoifnoexists{#1}%
758 {%
759 \bgroup
760 \let\@org@newglossaryentryprehook\@newglossaryentryprehook
761 \long\def\@newglossaryentryprehook{%
762 \long\def\@glo@desc{#3\glsxtrpostlongdescription}%
763 \@org@newglossaryentryprehook
764 }%
765 \renewcommand*\gls@assign@desc}[1]{%
766 \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%

```

The following is different from the base glossaries.sty:

```

767 \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
768 }
769 \gls@defglossaryentry{#1}{#2}%
770 \egroup
771 }%
772 }

```

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

```

773 \newcommand*\glsxtrpostlongdescription{\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.

```

774 \renewcommand{\newignoredglossary}{%
775 \@ifstar\glsxtr@s@newignoredglossary\glsxtr@org@newignoredglossary
776 }

```

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

```

777 \newcommand*\glsxtr@org@newignoredglossary}[1]{%
778 \ifcsdef{glolist@#1}
779 {%

```

```

780 \glstrundefaction{Glossary type ‘#1’ already exists}{}%
781 }%
782 {%
783 \ifdefempty\@ignored@glossaries
784 {%
785 \edef\@ignored@glossaries{#1}%
786 }%
787 {%
788 \eappto\@ignored@glossaries{,#1}%
789 }%
790 \csgdef{glolist@#1}{,}%
791 \ifcsundef{gls@#1@entryfmt}%
792 {%
793 \defglsentryfmt[#1]{\glsentryfmt}%
794 }%
795 }%
796 \ifdefempty\@gls@nohyperlist
797 {%
798 \renewcommand*{\@gls@nohyperlist}{#1}%
799 }%
800 {%
801 \eappto\@gls@nohyperlist{,#1}%
802 }%
803 }%
804 }

```

ignoredglossary Starred form.

```

805 \newcommand*{\glstr@s@newignoredglossary}[1]{%
806 \ifcsdef{glolist@#1}
807 {%
808 \glstrundefaction{Glossary type ‘#1’ already exists}{}%
809 }%
810 {%
811 \ifdefempty\@ignored@glossaries
812 {%
813 \edef\@ignored@glossaries{#1}%
814 }%
815 {%
816 \eappto\@ignored@glossaries{,#1}%
817 }%
818 \csgdef{glolist@#1}{,}%
819 \ifcsundef{gls@#1@entryfmt}%
820 {%
821 \defglsentryfmt[#1]{\glsentryfmt}%
822 }%
823 }%
824 }%
825 }

```

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

```
826 \glsifusetranslator
827 {%
828   \renewcommand*{\glssettoctitle}[1]{%
829     \ifcsdef{gls@tr@set@#1@toctitle}%
830     {%
831       \csuse{gls@tr@set@#1@toctitle}%
832     }%
833     {%
834       \ifcsdef{@glotype@#1@title}%
835       {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
836       {\def\glossarytoctitle{\glossarytitle}}%
837     }%
838   }%
839 }
840 {
841   \renewcommand*{\glssettoctitle}[1]{%
842     \ifcsdef{@glotype@#1@title}%
843     {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
844     {\def\glossarytoctitle{\glossarytitle}}%
845   }
846 }
```

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

```
847 \newcommand{\provideignoredglossary}{%
848   \@ifstar\glsxtr@s@provideignoredglossary\glsxtr@provideignoredglossary
849 }
```

`ignoredglossary` Unstarred version.

```
850 \newcommand*{\glsxtr@provideignoredglossary}[1]{%
851   \ifcsdef{glolist@#1}
852   {}%
853   {%
854     \ifdefempty\@ignored@glossaries
855     {%
856       \edef\@ignored@glossaries{#1}%
857     }%
858     {%
859       \eappto\@ignored@glossaries{,#1}%
860     }%
861     \csgdef{glolist@#1}{,}%
862     \ifcsundef{gls@#1@entryfmt}%
863     {%
864       \defglsentryfmt[#1]{\glsentryfmt}%
865     }%
866     {}%
867     \ifdefempty\@gls@nohyperlist
868     {%
```

```

869     \renewcommand*\@gls@nohyperlist}{#1}%
870   }%
871   {%
872     \eappto\@gls@nohyperlist{,#1}%
873   }%
874 }%
875 }

```

`ignoredglossary` Starred form.

```

876 \newcommand*\glsxtr@s@provideignoredglossary}[1]{%
877   \ifcsdef{glolist@#1}
878   {}%
879   {%
880     \ifdefempty\@ignored@glossaries
881     {%
882       \edef\@ignored@glossaries{#1}%
883     }%
884     {%
885       \eappto\@ignored@glossaries{,#1}%
886     }%
887     \csgdef{glolist@#1}{,}%
888     \ifcsundef{gls@#1@entryfmt}%
889     {%
890       \defglsentryfmt[#1]{\glsentryfmt}%
891     }%
892     {}%
893   }%
894 }

```

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

```

895 \newcommand*\glsxtrcopytoglossary}[2]{%
896   \glsdoifexists{#1}%
897   {%
898     \ifcsdef{glolist@#2}
899     {%
900       \cseappto{glolist@#2}{#1,}%
901     }%
902     {%
903       \glsxtrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
904     }%
905   }%
906 }

```

1.3.1 Existence Checks

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

```

907 \renewcommand*\glsdoifexists}[2]{%
908   \ifglsentryexists{#1}{#2}%

```

909 {%
Define `\glslabel` in case it's needed after this command (for example in the post-link hook).

```
910 \edef\glslabel{\glsdetoklabel{#1}}%  
911 \glstrundefaction{Glossary entry ‘\glslabel’  
912 has not been defined}{You need to define a glossary entry before  
913 you can reference it.}%  
914 }%  
915 }
```

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

```
916 \renewcommand{\glsdoifnoexists}[2]{%  
917 \ifglentryexists{#1}{%  
918 \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’  
919 has already been defined}{}}{#2}%  
920 }
```

`\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.

```
921 \ifdef\glsdoifexistsordo  
922 {%  
923 \renewcommand{\glsdoifexistsordo}[3]{%  
924 \ifglentryexists{#1}{#2}%  
925 {%  
926 \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’  
927 has not been defined}{You need to define a glossary entry  
928 before you can use it.}%  
929 #3%  
930 }%  
931 }%  
932 }  
933 {%  
934 \glstr@warnonexistsordo\glsdoifexistsordo  
935 \newcommand{\glsdoifexistsordo}[3]{%  
936 \ifglentryexists{#1}{#2}%  
937 {%  
938 \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’  
939 has not been defined}{You need to define a glossary entry  
940 before you can use it.}%  
941 #3%  
942 }%  
943 }%  
944 }
```

`\glsdoifglossarynoexistsordo` Similarly for `\doifglossarynoexistsordo`.

```
945 \ifdef\doifglossarynoexistsordo  
946 {%  
947 \renewcommand{\doifglossarynoexistsordo}[3]{%
```



```

948 \ifglossaryexists{#1}%
949 {%
950   \glstrundefaction{Glossary type ‘#1’ already exists}{}%
951   #3%
952   }%
953   {#2}%
954 }%
955 }
956 {%
957 \glstr@warnonexistsordo\doifglossarynoexistsordo
958 \newcommand{\doifglossarynoexistsordo}[3]{%
959   \ifglossaryexists{#1}%
960   {%
961     \glstrundefaction{Glossary type ‘#1’ already exists}{}%
962     #3%
963     }%
964     {#2}%
965   }%
966 }
967

```

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.

```

968 \appto\@newglossaryentryposthook{%
969   \ifdefvoid\@glo@see
970   {\csxdef{glo@\@glo@label @see}{}%
971     {%
972       \csxdef{glo@\@glo@label @see}{\@glo@see}%
973       \if@glstr@autoseeindex
974         \@glstr@autoindexcrossrefs
975       \fi
976     }%
977 }
978 \appto\@gls@keymap{,{see}{see}}

```

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

```

979 \newcommand*{\glstrusesee}[1]{%
980   \glsdoifexists{#1}%
981   {%
982     \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
983     \ifdefempty\@glo@see
984     {}%
985     {%
986       \expandafter\glstr@usesee\@glo@see\@end@glstr@usesee
987     }%
988   }%

```

```

989 }

\glxtr@usesee
990 \newcommand*{\glxtr@usesee}[1][\seename]{%
991   \@glxtr@usesee[#1]%
992 }

\@glxtr@usesee
993 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
994   \glxtruseseeformat{#1}{#2}%
995 }

xtruseseeformat   The format used by \glxtrusesee. The first argument is the tag (such as \seename). The
                  second argument is the comma-separated list of cross-referenced labels.
996 \newcommand*{\glxtruseseeformat}[2]{%
997   \glssееformat[#1]{#2}{}%
998 }

lsxtruseseealso  Apply \glssееformat to the seealso key if not empty. There's no optional tag to worry about
                  here.
999 \newcommand*{\glxtruseseealso}[1]{%
1000   \glsdоifexists{#1}%
1001   {%
1002     \letcs{\@glo@see}{glo@glsdetoklabel{#1}@seealso}%
1003     \ifdefempty\@glo@see
1004     {}%
1005     {%
1006       \expandafter\glxtruseseealsoformat\expandafter{\@glo@see}%
1007     }%
1008   }%
1009 }

sesealsoformat   The format used by \glxtruseseealso. The argument is the comma-separated list of
                  cross-referenced labels.
1010 \newcommand*{\glxtruseseealsoformat}[1]{%
1011   \glssееformat[\seealso]{#1}{}%
1012 }

\glxtrseelist    Fully expands argument before passing to \glssееlist. (The argument to \glssееlist
                  must be a comma-separated list of entry labels.)
1013 \newrobustcmd{\glxtrseelist}[1]{%
1014   \edef\@glo@tmp{\noexpand\glssееlist{#1}}\@glo@tmp
1015 }

\seealso         In case this command hasn't been defined. (Should be provided by language packages.)
1016 \providecommand{\seealso}{}

```

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

```
1017 \ifdef\@xdycrossrefhook
1018 {
```

Add the cross-reference class definition to the hook.

```
1019 \appto\@xdycrossrefhook{%
1020   \write\glswrite{(define-crossref-class \string"seealso\string"
1021     :unverified )}%
1022   \write\glswrite{(markup-crossref-list
1023     :class \string"seealso\string"^^J\space\space\space
1024     :open \string"\string\glsxtruseealsoformat\glsopenbrace\string"
1025     :close \string"\glsclosebrace\string")}%
1026 }
```

Append to class list.

```
1027 \appto\@dylocationclassorder{\space\string"seealso\string"}
```

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

```
1028 \newrobustcmd*{\glsxtrindexseealso}[2]{%
1029   \def\@gls@xref{#2}%
1030   \@onelevel@sanitize\@gls@xref
1031   \@gls@checkmkidxchars\@gls@xref
1032   \gls@glossary{\csname glo@#1@type\endcsname}{%
1033     (indexentry
1034       :tkey (\csname glo@#1@index\endcsname)
1035       :xref (\string"\@gls@xref\string")
1036       :attr \string"seealso\string"
1037     )
1038   }%
1039 }
```

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

```
1042 \newrobustcmd*{\glsxtrindexseealso}{\glssee[\seealsoname]}
1043 }
```

The alias key should be set to the label of the synonymous entry. The `seealso` key essentially behaves like `see=[\seealsoname]{\langle xr-list \rangle}`. 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`.

```
1044 \ifdef\gls@set@xr@key
1045 {
```

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

```

1046 \define@key{glossentry}{alias}{%
1047   \gls@set@xr@key{alias}{\@glo@alias}{#1}%
1048 }
1049 \define@key{glossentry}{seealso}{%
1050   \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
1051 }

```

Add to the key mappings.

```
1052 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}
```

Set the default value.

```
1053 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%
```

Assign the field values.

```

1054 \appto\@newglossaryentryposthook{%
1055   \ifdefvoid\@glo@seealso
1056     {\csxdef{glo@\@glo@label @seealso}{}}%
1057     {%
1058       \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1059       \if@glxtr@autoseeindex
1060         \@glxtr@autoindexcrossrefs
1061       \fi
1062     }%

```

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

```

1063   \ifdefvoid\@glo@alias
1064     {\csxdef{glo@\@glo@label @alias}{}}%
1065     {%
1066       \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1067     }%
1068 }

```

Provide user-level commands to access the values.

`\glxtralias`

```
1069 \newcommand*{\glxtralias}[1]{\@gls@entry@field{#1}{alias}}
```

`trseealsolabels`

```
1070 \newcommand*{\glxtrseealsolabels}[1]{\@gls@entry@field{#1}{seealso}}
```

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

```

1071 \appto\@glo@autoseehook{%
1072   \ifdefvoid\@glo@alias
1073     {%
1074       \ifdefvoid\@glo@seealso
1075         {}}%
1076     {%
1077       \edef\@do@glssee{\noexpand\glxtrindexseealso
1078         {\@glo@label}{\@glo@seealso}}%
1079       \@do@glssee

```

```

1080     }%
1081     }%
1082     {%

```

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

```

1083     \ifdefvoid\@glo@see
1084     {%
1085         \edef\@do@glssee{\noexpand\glssee{\@glo@label}{\@glo@alias}}%
1086         \@do@glssee
1087     }%
1088     {}%
1089     }%
1090 }%
1091 }
1092 {

```

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

`\glsxtralias`

```

1093 \glsaddstoragekey*{alias}{\glsxtralias}

```

`trseealsolabels`

```

1094 \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.

```

1095 \appto\@newglossaryentryposthook{%
1096     \ifcsvoid{glo@\@glo@label @alias}%
1097     {%
1098         \ifcsvoid{glo@\@glo@label @seealso}%
1099         {}%
1100         {%
1101             \edef\@do@glssee{\noexpand\glsxtrindexseealso
1102                 {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1103             \@do@glssee
1104         }%
1105     }%
1106     {%

```

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

```

1107     \ifdefvoid\@glo@see
1108     {%
1109         \edef\@do@glssee{\noexpand\glssee
1110             {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1111         \@do@glssee
1112     }%
1113     {}%
1114     }%
1115 }

```

1116 }

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

1117 \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.

1118 \newcommand*{\glxtraddallcrossrefs}{%

1119 \foralllglossaries{\@glo@type}%

1120 {%

1121 \forglsentries[\@glo@type]{\@glo@label}%

1122 {%

1123 \ifglsused{\@glo@label}%

1124 {\expandafter\@glxtr@addunusedxrefs\expandafter{\@glo@label}}{}}%

1125 }%

1126 }%

1127 }

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

1128 \newcommand*{\@glxtr@addunusedxrefs}[1]{%

1129 \letcs{\@glo@see}{glo\glsdetoklabel{#1}@see}%

1130 \ifdefvoid\@glo@see

1131 {}%

1132 {%

1133 \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused

1134 }%

1135 \letcs{\@glo@see}{glo\glsdetoklabel{#1}@seealso}%

1136 \ifdefvoid\@glo@see

1137 {}%

1138 {%

1139 \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused

1140 }%

1141 }

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

1142 \newcommand*{\glxtr@addunused}[1][[]]{%

1143 \@glxtr@addunused

1144 }

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

1145 \def\@glxtr@addunused#1\@end@glxtr@addunused{%

1146 \@for\@glxtr@label:=#1\do

1147 {%

1148 \ifglsused{\@glxtr@label}}{}}%

1149 {%

1150 \glsadd[format=glxtrunusedformat]{\@glxtr@label}%

1151 \glsunset{\@glxtr@label}%

1152 \expandafter\@glxtr@addunusedxrefs\expandafter{\@glxtr@label}}%

```

1153 }%
1154 }%
1155 }

```

xtrunusedformat

```

1156 \newcommand*{\glxtrunusedformat}[1]{\unskip}

```

1.3.2 Document Definitions

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

```

1157 \let\glxtr@orgmakenoidxglossaries\makenoidxglossaries
1158 \renewcommand{\makenoidxglossaries}{%
1159 \glxtr@orgmakenoidxglossaries
1160 \if@glxtrdocdefrestricted

```

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

```

1161 \renewcommand*{\@gls@reference}[3]{%
1162 \ifcsundef{@glsref###1}{\csgdef{@glsref###1}{}}{%
1163 \ifinlistcs{##2}{@glsref###1}%
1164 }{%
1165 {\listcsgadd{@glsref###1}{##2}}%
1166 \ifcsundef{glo@glstdetoklabel{##2}@loclist}%
1167 {\csgdef{glo@glstdetoklabel{##2}@loclist}{}}%
1168 }%
1169 \listcsgadd{glo@glstdetoklabel{##2}@loclist}{##3}%
1170 }%
1171 \else

```

Disable document definitions.

```

1172 \@glxtrdocdeffalse
1173 \fi
1174 \disable@keys{glossaries-extra.sty}{docdef}%
1175 }

```

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

```

1176 \renewcommand*{\gls@defdocnewglossaryentry}{%
1177 \ifcase\@glxtr@docdefval
1178 docdef=false:
1179 \renewcommand*{\newglossaryentry}[2]{%
1180 \PackageError{glossaries-extra}{Glossary entries must
1181 be \MessageBreak defined in the preamble with \MessageBreak
1182 package option 'docdef=false'\MessageBreak(consider using
1183 'docdef=restricted')}{Move your glossary definitions to
1184 the preamble. You can also put them in a \MessageBreak separate file
1185 and load them with \string\loadglsentries.}%
1186 }%
1187 \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.

```
1187 \let\gls@checkseeallowed\relax
1188 \let\newglossaryentry\new@glossaryentry
1189 \or
```

Restricted mode just needs to allow the see value.

```
1190 \let\gls@checkseeallowed\relax
1191 \fi
1192 }%
```

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.

`\GlsXtrEnableOnTheFly`

```
1193 \newcommand*{\GlsXtrEnableOnTheFly}{%
1194 \ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
1195 }
```

`\@sGlsXtrEnableOnTheFly`

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.)

```
1196 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
1197 \renewcommand*{\glsdetoklabel}[1]{%
1198 \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
1199 {%
1200 \expandafter\detokenize\expandafter{##1}%
1201 }%
1202 {\detokenize{##1}}%
1203 }%
1204 \@GlsXtrEnableOnTheFly
1205 }
1206 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
1207 \expandafter\if\glsbackslash#1%
1208 #3%
1209 \else
1210 #4%
1211 \fi
1212 }
```

`\GlsXtrStarflywarn`

```
1213 \newcommand*{\GlsXtrStarflywarn}{%
1214 \GlossariesExtraWarning{Experimental starred version of
1215 \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
1216 read the warnings in the glossaries-extra user manual)}}%
1217 }
```


rEnableOnTheFly

```
1218 \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`

```
1219 \newcommand*{\glsxtrcat}{general}
```

`\glsxtr`

```
1220 \newcommand*{\glsxtr}[1] [] {%
```

```
1221 \def\glsxtr@keylist{##1}%
```

```
1222 \@glsxtr
```

```
1223 }
```

`\@glsxtr`

```
1224 \newcommand*{\@glsxtr}[2] [] {%
```

```
1225 \ifglsentryexists{##2}%
```

```
1226 {%
```

```
1227 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
```

```
1228 }%
```

```
1229 {%
```

```
1230 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
```

```
1231 description={\nopostdesc},##1}%
```

```
1232 }%
```

```
1233 \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
```

```
1234 }
```

`\Glsxtr`

```
1235 \newcommand*{\Glsxtr}[1] [] {%
```

```
1236 \def\glsxtr@keylist{##1}%
```

```
1237 \@Glsxtr
```

```
1238 }
```

`\@Glsxtr`

```
1239 \newcommand*{\@Glsxtr}[2] [] {%
```

```
1240 \ifglsentryexists{##2}%
```

```
1241 {%
```

```
1242 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
```

```
1243 }%
```

```
1244 {%
```

```
1245 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
```

```
1246 description={\nopostdesc},##1}%
```

```
1247 }%
```

```
1248 \expandafter\Gls\expandafter[\glsxtr@keylist]{##2}%
```

```
1249 }
```

`\glsxtrpl`

```
1250 \newcommand*\glsxtrpl[1][]{%
1251 \def\glsxtr@keylist{##1}%
1252 \@glsxtrpl
1253 }
```

`\@glsxtrpl`

```
1254 \newcommand*\@glsxtrpl[2][]{%
1255 \ifglsentryexists{##2}%
1256 {%
1257 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
1258 }%
1259 {%
1260 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1261 description={\nopostdesc},##1}%
1262 }%
1263 \expandafter\glspl\expandafter[\glsxtr@keylist]{##2}%
1264 }
```

`\Glsxtrpl`

```
1265 \newcommand*\Glsxtrpl[1][]{%
1266 \def\glsxtr@keylist{##1}%
1267 \@Glsxtrpl
1268 }
```

`\@Glsxtrpl`

```
1269 \newcommand*\@Glsxtrpl[2][]{%
1270 \ifglsentryexists{##2}
1271 {%
1272 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
1273 }%
1274 {%
1275 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1276 description={\nopostdesc},##1}%
1277 }%
1278 \expandafter\Glspl\expandafter[\glsxtr@keylist]{##2}%
1279 }
```

`\GlsXtrWarning`

```
1280 \newcommand*\GlsXtrWarning[2]{%
1281 \def\@glsxtr@optlist{##1}%
1282 \@onelevel@sanitize\@glsxtr@optlist
1283 \GlossariesExtraWarning{The options ‘\@glsxtr@optlist’ have
1284 been ignored for entry ‘##2’ as it has already been defined}%
1285 }
```

Disable commands after the glossary:

```
1286 \renewcommand\@printglossary[2]{%
```

```

1287 \def\@glsxtr@printglossopts{##1}%
1288 \@glsxtr@orgprintglossary{##1}{##2}%
1289 \def\@glsxtr{\@glsxtr@disabledflycommand\glsxtr}%
1290 \def\@glsxtrpl{\@glsxtr@disabledflycommand\glsxtrpl}%
1291 \def\@Glsxtr{\@glsxtr@disabledflycommand\Glsxtr}%
1292 \def\@Glsxtrpl{\@glsxtr@disabledflycommand\Glsxtrpl}%
1293 }

```

abledflycommand

```

1294 \newcommand*{\@glsxtr@disabledflycommand}[1]{%
1295 \PackageError{glossaries-extra}%
1296 {\string##1\space can't be used after any of the \MessageBreak
1297 glossaries have been displayed}%
1298 {The on-the-fly commands enabled by
1299 \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1300 before the glossaries. If you want to use any entries \MessageBreak
1301 after any of the glossaries, you must use the standard \MessageBreak
1302 method of first defining the entry and then using the \MessageBreak
1303 entry with commands like \string\gls}%
1304 \@glsxtr@disabledflycommand
1305 }%
1306 \newcommand*{\@glsxtr@disabledflycommand}[2][{}]{##2}

```

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

```

1307 \let\GlsXtrEnableOnTheFly\relax
1308 }
1309 \@onlypreamble\GlsXtrEnableOnTheFly

```

1.3.3 Existing Glossary Style Modifications

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

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

```

1310 \newcommand*{\@glsxtr@current@style}{\@glossary@default@style}

```

Modify \setglossarystyle to set \@glsxtr@current@style.

etglossarystyle

```

1311 \renewcommand*{\setglossarystyle}[1]{%
1312 \ifcsundef{@glsstyle@#1}%
1313 {%
1314 \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%
1315 }%
1316 {%
1317 \csname @glsstyle@#1\endcsname

```

Only set the current style if it exists.

```

1318 \protected@edef\@glsxtr@current@style{#1}%
1319 }%

```

```

1320 \ifx@glossary@default@style\relax
1321   \protected@edef@glossary@default@style{#1}%
1322 \fi
1323 }

```

In case we have an old version of glossaries:

```

1324 \ifdef@glossary@default@style
1325 {}
1326 {%
1327   \let@glossary@default@style\relax
1328 }

```

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

```

1329 \ifdef\glslistdottedwidth
1330 {%
1331   \ifdim\glslistdottedwidth=.5\hsize
1332     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1333     \AtBeginDocument{%
1334       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1335         \setlength{\glslistdottedwidth}{.5\columnwidth}%
1336       \fi
1337     }%
1338   \fi
1339 }
1340 {}%

```

Similarly for `\glsdescwidth`:

`\glsdescwidth`

```

1341 \ifdef\glsdescwidth
1342 {%
1343   \ifdim\glsdescwidth=.6\hsize
1344     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1345     \AtBeginDocument{%
1346       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1347         \setlength{\glsdescwidth}{.6\columnwidth}%
1348       \fi
1349     }%
1350   \fi
1351 }
1352 {}%

```

and for `\glspagelistwidth`:

`lspagelistwidth`

```

1353 \ifdef\glspagelistwidth
1354 {%
1355   \ifdim\glspagelistwidth=.1\hsize
1356     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}

```

```

1357 \AtBeginDocument{%
1358     \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1359     \setlength{\glspagelistwidth}{.1\columnwidth}%
1360     \fi
1361 }%
1362 \fi
1363 }
1364 {}%

```

aryentrynumbers Has the nonnumberlist option been used?

```

1365 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1366 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1367   \glsnonnumberlistfalse
1368   \renewcommand*{\glossaryentrynumbers}[1]{%
1369     \ifglsentryexists{\glscurrententrylabel}%
1370     {%
1371       \@glsxtrpreloctag
1372       \GlsXtrFormatLocationList{#1}%
1373       \@glsxtrpostloctag
1374       \gls@save@numberlist{#1}%
1375     }{}%
1376   }%
1377 \else
1378   \glsnonnumberlisttrue
1379   \renewcommand*{\glossaryentrynumbers}[1]{%
1380     \ifglsentryexists{\glscurrententrylabel}%
1381     {%
1382       \gls@save@numberlist{#1}%
1383     }{}%
1384   }%
1385 \fi

```

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

```

1386 \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

```

1387 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
1388   \let\@glsxtrpreloctag\@glsxtrpreloctag
1389   \let\@glsxtrpostloctag\@glsxtrpostloctag
1390   \renewcommand*{\@glsxtr@pagetag}{#1}%
1391   \renewcommand*{\@glsxtr@pagestag}{#2}%
1392   \renewcommand*{\@glsxtr@savepreloctag}[2]{%

```

```

1393 \csgdef{@glxtr@preloctag@##1}{##2}%
1394 }%
1395 \renewcommand*{\@glxtr@doloctag}{%
1396 \ifcsundef{@glxtr@preloctag@glscurrententrylabel}%
1397 {%
1398 \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
1399 Rerun required}%
1400 }%
1401 {%
1402 \csuse{@glxtr@preloctag@glscurrententrylabel}%
1403 }%
1404 }%
1405 }
1406 \@onlypreamble\GlsXtrEnablePreLocationTag

```

glxtrpreloctag

```

1407 \newcommand*{\@glxtrpreloctag}{%
1408 \let\@glxtr@org@delimN\delimN
1409 \let\@glxtr@org@delimR\delimR
1410 \let\@glxtr@org@glsgignore\glsgignore
1411 \gdef is required as the delimiters may occur inside a scope.
1412 \gdef \@glxtr@thisloctag{\@glxtr@pagetag}%
1413 \renewcommand*{\delimN}{%
1414 \gdef \@glxtr@thisloctag{\@glxtr@pagetag}%
1415 \@glxtr@org@delimN}%
1416 \renewcommand*{\delimR}{%
1417 \gdef \@glxtr@thisloctag{\@glxtr@pagetag}%
1418 \@glxtr@org@delimR}%
1419 \renewcommand*{\glsgignore}[1]{%
1420 \gdef \@glxtr@thisloctag{\relax}%
1421 \@glxtr@org@glsgignore{##1}}%
1422 \@glxtr@doloctag
1422 }

```

glxtrpreloctag

```

1423 \newcommand*{\@glxtrpreloctag}{}

```

@glxtr@pagetag

```

1424 \newcommand*{\@glxtr@pagetag}{}%

```

glxtr@pagetag

```

1425 \newcommand*{\@glxtr@pagetag}{}%

```

lxstrpostloctag

```

1426 \newcommand*{\@glxtrpostloctag}{%
1427 \let\delimN\@glxtr@org@delimN
1428 \let\delimR\@glxtr@org@delimR
1429 \let\glsgignore\@glxtr@org@glsgignore

```

```

1430 \protected@write\@auxout{}%
1431 {\string\@glxtr@savepreloctag\glscurrententrylabel}\@glxtr@thisloctag}}%
1432 }

```

lsxtrpostloctag

```

1433 \newcommand*\@glxtrpostloctag{}

```

lsxtr@preloctag

```

1434 \newcommand*\@glxtr@savepreloctag}[2]{}
1435 \protected@write\@auxout{}{%
1436 \string\providecommand\string\@glxtr@savepreloctag[2]{}
}

```

glxtr@doloctag

```

1437 \newcommand*\@glxtr@doloctag{}

```

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

```

1438 \renewcommand*\KV@printgloss@nonumberlist}[1]{}%
1439 \XKV@plfalse
1440 \XKV@strtrue
1441 \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
1442 {}%
1443 \csname glsnonumberlist\XKV@resa\endcsname
1444 \ifglsglossaryentrynumbers
1445 \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1446 \else
1447 \def\glossaryentrynumbers##1{%
1448 \@glxtrpreloctag
1449 \GlsXtrFormatLocationList{##1}%
1450 \@glxtrpostloctag
1451 \gls@save@numberlist{##1}}%
1452 \fi
1453 }%
1454 }

```

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.

```

1455 \renewcommand*\glsentryfmt{}%
1456 \ifglshasshort{\glslabel}\glssetabbrfmt{\glscategory{\glslabel}}{}%
1457 \glsifregular{\glslabel}%
1458 {\glsxtrregularfont\glsgenentryfmt}}%

```

```

1459  {%
1460    \ifglshasshort{\glslabel}%
1461    {\glsxtrgenabbrvfmt}%
1462    {\glsxtrregularfont{\glsgenentryfmt}}%
1463  }%
1464 }

```

`sxtrregularfont` Font used for regular entries.

```
1465 \newcommand*{\glsxtrregularfont}[1]{#1}
```

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

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

```
1466 \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.

```

1467  \@glxtr@record{#2}{#3}{glslink}%
1468  \glsdoifexists{#3}%
1469  {%

```

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).

```

1470  \let\glxtrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
1471  \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1472  \def\glscustomtext{#4}%
1473  \@glxtr@field@linkdefs
1474  #1%
1475  \@gls@link[#2]{#3}{#4}%
1476  \let\ifKV@glslink@hyper\glxtrorg@ifKV@glslink@hyper
1477  }%
1478  \glspostlinkhook
1479 }

```

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

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

```

1480 \let\@glxtr@org@gls@\@gls@
1481 \def\@gls@#1#2{%
1482  \@glxtr@record{#1}{#2}{glslink}%
1483  \@glxtr@org@gls@{#1}{#2}%
1484 }%

```


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

```
1485 \let\@glsxtr@org@glspl@\@glspl@
1486 \def\@glspl@#1#2{%
1487   \@glsxtr@record{#1}{#2}{glslink}%
1488   \@glsxtr@org@glspl@{#1}{#2}%
1489 }%
```

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

```
1490 \let\@glsxtr@org@Gls@\@Gls@
1491 \def\@Gls@#1#2{%
1492   \@glsxtr@record{#1}{#2}{glslink}%
1493   \@glsxtr@org@Gls@{#1}{#2}%
1494 }%
```

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

```
1495 \let\@glsxtr@org@Glspl@\@Glspl@
1496 \def\@Glspl@#1#2{%
1497   \@glsxtr@record{#1}{#2}{glslink}%
1498   \@glsxtr@org@Glspl@{#1}{#2}%
1499 }%
```

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

```
1500 \let\@glsxtr@org@GLS@\@GLS@
1501 \def\@GLS@#1#2{%
1502   \@glsxtr@record{#1}{#2}{glslink}%
1503   \@glsxtr@org@GLS@{#1}{#2}%
1504 }%
```

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

```
1505 \let\@glsxtr@org@GLSpl@\@GLSpl@
1506 \def\@GLSpl@#1#2{%
1507   \@glsxtr@record{#1}{#2}{glslink}%
1508   \@glsxtr@org@GLSpl@{#1}{#2}%
1509 }%
```

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

```
1510 \renewcommand*{\@glsdisp}[3][[]]{%
1511   \@glsxtr@record{#1}{#2}{glslink}%
1512   \glsdoifexists{#2}{%
1513     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
1514     \let\glsifplural\@secondoftwo
1515     \let\glsapscase\@firstofthree
1516     \def\glscustomtext{#3}%
1517     \def\glsinsert{}%
1518     \def\@glo@text{\csname gls@\gls@type @entryfmt\endcsname}%
1519     \@gls@link[#1]{#2}{\@glo@text}%
1520     \ifKV@glslink@local
```

```

1521     \glslocalunset{#2}%
1522   \else
1523     \glsunset{#2}%
1524   \fi
1525 }%
1526 \glspostlinkhook
1527 }

```

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

```

1528 \renewcommand*{\@gls@link}[3][\%
1529   \@glsxtr@record{#1}{#2}{glslink}%
1530   \glsdoifexistsordo{#2}%
1531   {%
1532     \let\do@gls@link@checkfirsthyper\relax
1533     \@gls@link[#1]{#2}{#3}%
1534   }%
1535   {%
1536     \glstextformat{#3}%
1537   }%
1538   \glspostlinkhook
1539 }

```

`sxtrinitwrgloss` Set the default if the wrgloss is omitted.

```

1540 \newcommand*{\glsxtrinitwrgloss}{%
1541   \glsifattribute{\glslabel}{wrgloss}{after}%
1542   {%
1543     \glsxtrinitwrglossbeforefalse
1544   }%
1545   {%
1546     \glsxtrinitwrglossbeforetrue
1547   }%
1548 }

```

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

```

1549 \newif\ifglsxtrinitwrglossbefore
1550 \glsxtrinitwrglossbeforetrue

```

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

```

1551 \define@choicekey{glslink}{wrgloss}[\val\nr]{before,after}%
1552 {%
1553   \ifcase\nr\relax
1554     \glsxtrinitwrglossbeforetrue
1555   \or
1556     \glsxtrinitwrglossbeforefalse
1557   \fi
1558 }

```

```

1559 \define@key{glslink}{thevalue}{\def\@glsxtr@thevalue{#1}}

```

```
1560 \define@key{glslink}{theHvalue}{\def\@glsxtr@theHvalue{#1}}
```

`\@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.

```
1561 \def\@gls@link[#1]#2#3{%
1562   \leavevmode
1563   \edef\glslabel{\glsdetoklabel{#2}}%
1564   \def\@gls@link@opts{#1}%
1565   \let\@gls@link@label\glslabel
1566   \let\@glsnumberformat\@glsxtr@defaultnumberformat
1567   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
1568   \edef\glsstype{\csname glo@\glslabel @type\endcsname}%
1569   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
```

Initialise thevalue and theHvalue (v1.19).

```
1570 \def\@glsxtr@thevalue{}%
1571 \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
```

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

```
1572 \glsxtrinitwrgloss
```

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

```
1573 \@gls@setdefault@glslink@opts
1574 \do@gls@disablehyperinlist
1575 \do@gls@link@checkfirsthyper
1576 \setkeys{glslink}{#1}%
1577 \glslinkpostsetkeys
```

Check thevalue and theHvalue before saving (v1.19).

```
1578 \ifdefempty{\@glsxtr@thevalue}%
1579 {%
1580   \@gls@saveentrycounter
1581 }%
1582 {%
1583   \let\thegl@entrycounter\@glsxtr@thevalue
1584   \def\theHgl@entrycounter{\@glsxtr@theHvalue}%
1585 }%
1586 \@gls@setsort{\glslabel}%
```

Do write if it should occur before the link text:

```
1587 \ifglsxtrinitwrglossbefore
1588   \do@wrglossary{#2}%
1589 \fi
```

Do the link text:

```
1590 \ifKV@glslink@hyper
1591   \@glslink{\glolinkprefix\glslabel}{\gls@textformat{#3}}%
1592 \else
1593   \glsdonohyperlink{\glolinkprefix\glslabel}{\gls@textformat{#3}}%
1594 \fi
```

Do write if it should occur after the link text:

```
1595 \ifglxtrinitwrglossbefore
1596 \else
1597   \@do@wrglossary{#2}%
1598 \fi
```

As the original definition:

```
1599 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
1600 }
```

```
1601 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}
```

```
1602 \define@key{glossadd}{theHvalue}{\def\@glxtr@theHvalue{#1}}
```

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

```
1603 \renewrobustcmd*{\glsadd}[2][ ]{%
1604   \@gls@adjustmode
1605   \@glxtr@record{#1}{#2}{glossadd}%
1606   \glsdoifexists{#2}%
1607   {%
1608     \let\@glsnumberformat\@glxtr@defaultnumberformat
1609     \edef\@gls@counter{\csname glo@\glsdetoklabel{#2}@counter\endcsname}%
1610     \def\@glxtr@thevalue{}%
1611     \def\@glxtr@theHvalue{\@glxtr@thevalue}%
1612     \setkeys{glossadd}{#1}%
1613     \ifdefempty{\@glxtr@thevalue}%
1614     {%
1615       \@gls@saveentrycounter
1616     }%
1617     {%
1618       \let\theglentrycounter\@glxtr@thevalue
1619       \def\theHglentrycounter{\@glxtr@theHvalue}%
1620     }%
1621     \@do@wrglossary{#2}%
1622   }%
1623 }
```

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

```
1624 \newcommand*{\@glxtr@field@linkdefs}{%
1625   \let\glxtrifwasfirstuse\@secondoftwo
1626   \let\glsifplural\@secondoftwo
1627   \let\glscapscase\@firstofthree
1628   \let\glsinsert\@empty
1629 }
```

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

assignfieldfont

```
1630 \newcommand*{\glstrassignfieldfont}[1]{%
1631   \ifglstryexists{#1}%
1632   {%
1633     \ifglshasshort{#1}%
1634     {%
1635       \glsetabbrvfmt{\glscategory{#1}}%
1636       \glrifregular{#1}%
1637       {\let\@gls@field@font\glstrregularfont}%
1638       {\let\@gls@field@font\@firstofone}%
1639     }%
1640   }%
1641   \glrifnotregular{#1}%
1642   {\let\@gls@field@font\@firstofone}%
1643   {\let\@gls@field@font\glstrregularfont}%
1644 }%
1645 }%
1646 {%
1647   \let\@gls@field@font\@gobble
1648 }%
1649 }
```

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

```
1650 \def\@glstext@#1#2[#3]{%
1651   \glstrassignfieldfont{#2}%
1652   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
1653 }
```

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

```
1654 \def\@GLStext@#1#2[#3]{%
1655   \glstrassignfieldfont{#2}%
1656   \@gls@field@link[\let\glscaps@case\@thirdofthree]{#1}{#2}%
1657   {\@gls@field@font{\GLS@accesstext{#2}\mfirstucMakeUppercase{#3}}}%
1658 }
```

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

```
1659 \def\@Glstext@#1#2[#3]{%
1660   \glstrassignfieldfont{#2}%
1661   \@gls@field@link[\let\glscaps@case\@secondofthree]{#1}{#2}%
1662   {\@gls@field@font{\GLS@accesstext{#2}#3}}%
1663 }
```

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.

checknohyperfirst

```
1664 \newcommand*{\glstrchecknohyperfirst}[1]{%
1665   \glrifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
1666 }
```

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

```
1667 \def\@glsfirst@#1#2[#3]{%
1668   \glstrassignfieldfont{#2}%
      Ensure that \glsfirst honours the nohyperfirst attribute.
1669   \@gls@field@link
1670   [\let\glstrifwasfirstuse\@firstoftwo
1671     \glstrchecknohyperfirst{#2}%
1672   ]{#1}{#2}%
1673   {\@gls@field@font{\glsaccessfirst{#2}#3}}%
1674 }
```

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

```
1675 \def\@Glsfirst@#1#2[#3]{%
1676   \glstrassignfieldfont{#2}%
      Ensure that \Glsfirst honours the nohyperfirst attribute.
1677   \@gls@field@link
1678   [\let\glstrifwasfirstuse\@firstoftwo
1679     \let\glscapscase\@secondofthree
1680     \glstrchecknohyperfirst{#2}%
1681   ]%
1682   {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
1683 }
```

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

```
1684 \def\@GLSfirst@#1#2[#3]{%
1685   \glstrassignfieldfont{#2}%
      Ensure that \GLSfirst honours the nohyperfirst attribute.
1686   \@gls@field@link
1687   [\let\glstrifwasfirstuse\@firstoftwo
1688     \let\glscapscase\@thirdofthree
1689     \glstrchecknohyperfirst{#2}%
1690   ]%
1691   {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1692 }
```

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

```
1693 \def\@glsplural@#1#2[#3]{%
1694   \glstrassignfieldfont{#2}%
1695   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1696   {\@gls@field@font{\glsaccessplural{#2}#3}}%
1697 }
```

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

```
1698 \def\@Glsplural@#1#2[#3]{%
1699   \glstrassignfieldfont{#2}%
1700   \@gls@field@link
1701   [\let\glsifplural\@firstoftwo
```

```

1702 \let\gls caps case \@secondofthree
1703 ]%
1704 {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1705 }

```

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

```

1706 \def\@GLSplural@#1#2[#3]{%
1707 \glsxtrassignfieldfont{#2}%
1708 \@gls@field@link
1709 [\let\glsifplural\@firstoftwo
1710 \let\gls caps case \@thirdofthree
1711 ]%
1712 {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1713 }

```

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

```

1714 \def\@glsfirstplural@#1#2[#3]{%
1715 \glsxtrassignfieldfont{#2}%
1716 \gls@field@link
1717 [\let\glsxtrifwasfirstuse\@firstoftwo
1718 \let\glsifplural\@firstoftwo
1719 \glsxtrchecknohyperfirst{#2}%
1720 ]%
1721 {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1722 }

```

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

```

1723 \def\@Glsfirstplural@#1#2[#3]{%
1724 \glsxtrassignfieldfont{#2}%
1725 \gls@field@link
1726 [\let\glsxtrifwasfirstuse\@firstoftwo
1727 \let\glsifplural\@firstoftwo
1728 \let\gls caps case \@secondofthree
1729 \glsxtrchecknohyperfirst{#2}%
1730 ]%
1731 {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1732 }

```

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

```

1733 \def\@GLSfirstplural@#1#2[#3]{%
1734 \glsxtrassignfieldfont{#2}%
1735 \gls@field@link
1736 [\let\glsxtrifwasfirstuse\@firstoftwo
1737 \let\glsifplural\@firstoftwo

```

```

1738 \let\glscapscase\@thirdofthree
1739 \glsxtrchecknohyperfirst{#2}%
1740 ]%
1741 {#1}{#2}%
1742 {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
1743 }

```

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

```

1744 \def\@glsname@#1#2[#3]{%
1745 \glsxtrassignfieldfont{#2}%
1746 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
1747 }

```

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

```

1748 \def\@Glsname@#1#2[#3]{%
1749 \glsxtrassignfieldfont{#2}%
1750 \@gls@field@link
1751 [\let\glsapspace\@secondoftwo]{#1}{#2}%
1752 {\@gls@field@font{\GLSaccessname{#2}#3}}%
1753 }

```

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

```

1754 \def\@GLSname@#1#2[#3]{%
1755 \glsxtrassignfieldfont{#2}%
1756 \@gls@field@link[\let\glsapspace\@thirdoftwo]%
1757 {#1}{#2}%
1758 {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
1759 }

```

`\@glsdesc@`

```

1760 \def\@glsdesc@#1#2[#3]{%
1761 \glsxtrassignfieldfont{#2}%
1762 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
1763 }

```

`\@Glsdesc@` First letter uppercase version.

```

1764 \def\@Glsdesc@#1#2[#3]{%
1765 \glsxtrassignfieldfont{#2}%
1766 \@gls@field@link
1767 [\let\glsapspace\@secondoftwo]{#1}{#2}%
1768 {\@gls@field@font{\GLSaccessdesc{#2}#3}}%
1769 }

```

`\@GLSdesc@` All uppercase version.

```

1770 \def\@GLSdesc@#1#2[#3]{%
1771 \glsxtrassignfieldfont{#2}%
1772 \@gls@field@link[\let\glsapspace\@thirdoftwo]%
1773 {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
1774 }

```


@glsdescplural@ No case-changing version.

```
1775 \def@glsdescplural@#1#2[#3]{%
1776   \glstrassignfieldfont{#2}%
1777   \@gls@field@link
1778   [\let\glscaps@case@secondoftwo
1779   \let\glsifplural@firstoftwo
1780   ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
1781 }
```

@Glsdescplural@ First letter uppercase version.

```
1782 \def@Glsdescplural@#1#2[#3]{%
1783   \glstrassignfieldfont{#2}%
1784   \@gls@field@link
1785   [\let\glscaps@case@secondoftwo
1786   \let\glsifplural@firstoftwo
1787   ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
1788 }
```

@GLSdescplural@ All uppercase version.

```
1789 \def@GLSdesc@#1#2[#3]{%
1790   \glstrassignfieldfont{#2}%
1791   \@gls@field@link
1792   [\let\glscaps@case@thirdoftwo
1793   \let\glsifplural@firstoftwo
1794   ]%
1795   {#1}{#2}%
1796   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
1797 }
```

\@glssymbol@

```
1798 \def@glssymbol@#1#2[#3]{%
1799   \glstrassignfieldfont{#2}%
1800   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
1801 }
```

\@Glsymbol@ First letter uppercase version.

```
1802 \def@Glsymbol@#1#2[#3]{%
1803   \glstrassignfieldfont{#2}%
1804   \@gls@field@link
1805   [\let\glscaps@case@secondoftwo]%
1806   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
1807 }
```

\@GLSsymbol@ All uppercase version.

```
1808 \def@GLSsymbol@#1#2[#3]{%
1809   \glstrassignfieldfont{#2}%
1810   \@gls@field@link[\let\glscaps@case@thirdoftwo]%
1811   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
1812 }
```

lssymbolplural@ No case-changing version.

```
1813 \def\@glssymbolplural@#1#2[#3]{%
1814   \glstrassignfieldfont{#2}%
1815   \@gls@field@link
1816   [\let\glscapscase\@secondoftwo
1817   \let\glsifplural\@firstoftwo
1818   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
1819 }
```

lssymbolplural@ First letter uppercase version.

```
1820 \def\@Glsymbolplural@#1#2[#3]{%
1821   \glstrassignfieldfont{#2}%
1822   \@gls@field@link
1823   [\let\glscapscase\@secondoftwo
1824   \let\glsifplural\@firstoftwo
1825   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
1826 }
```

LSsymbolplural@ All uppercase version.

```
1827 \def\@GLSsymbol@#1#2[#3]{%
1828   \glstrassignfieldfont{#2}%
1829   \@gls@field@link
1830   [\let\glscapscase\@thirdoftwo
1831   \let\glsifplural\@firstoftwo
1832   ]%
1833   {#1}{#2}%
1834   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
1835 }
```

\@Glsuseri@ First letter uppercase version.

```
1836 \def\@Glsuseri@#1#2[#3]{%
1837   \glstrassignfieldfont{#2}%
1838   \@gls@field@link
1839   [\let\glscapscase\@secondoftwo]{#1}{#2}%
1840   {\@gls@field@font{\Glsentryuseri{#2}#3}}%
1841 }
```

\@GLSuseri@ All uppercase version.

```
1842 \def\@GLSuseri@#1#2[#3]{%
1843   \glstrassignfieldfont{#2}%
1844   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1845   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}}%
1846 }
```

\@Glsuserii@ First letter uppercase version.

```
1847 \def\@Glsuserii@#1#2[#3]{%
1848   \glstrassignfieldfont{#2}%
1849   \@gls@field@link
```

```

1850 [\let\glscapscase\@secondoftwo]%
1851   {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
1852 }

```

`\@GLSuserii@` All uppercase version.

```

1853 \def\@GLSuserii@#1#2[#3]{%
1854   \glstrassignfieldfont{#2}%
1855   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1856   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
1857 }

```

`\@Glsuseriii@` First letter uppercase version.

```

1858 \def\@Glsuseriii@#1#2[#3]{%
1859   \glstrassignfieldfont{#2}%
1860   \@gls@field@link
1861   [\let\glscapscase\@secondoftwo]%
1862   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
1863 }

```

`\@GLSuseriii@` All uppercase version.

```

1864 \def\@GLSuseriii@#1#2[#3]{%
1865   \glstrassignfieldfont{#2}%
1866   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1867   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
1868 }

```

`\@Glsuseriv@` First letter uppercase version.

```

1869 \def\@Glsuseriv@#1#2[#3]{%
1870   \glstrassignfieldfont{#2}%
1871   \@gls@field@link
1872   [\let\glscapscase\@secondoftwo]%
1873   {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
1874 }

```

`\@GLSuseriv@` All uppercase version.

```

1875 \def\@GLSuseriv@#1#2[#3]{%
1876   \glstrassignfieldfont{#2}%
1877   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1878   {#1}{#2}%
1879   {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
1880 }

```

`\@Glsuserv@` First letter uppercase version.

```

1881 \def\@Glsuserv@#1#2[#3]{%
1882   \glstrassignfieldfont{#2}%
1883   \@gls@field@link
1884   [\let\glscapscase\@secondoftwo]%
1885   {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
1886 }

```

`\@GLSuserv@` All uppercase version.

```
1887 \def\@GLSuserv@#1#2[#3]{%
1888   \glstrassignfieldfont{#2}%
1889   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1890   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glstentryserv{#2}#3}}}%
1891 }
```

`\@Glsuservi@` First letter uppercase version.

```
1892 \def\@Glsuservi@#1#2[#3]{%
1893   \glstrassignfieldfont{#2}%
1894   \@gls@field@link
1895   [\let\glscapscase\@secondoftwo]%
1896   {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}%
1897 }
```

`\@GLSuservi@` All uppercase version.

```
1898 \def\@GLSuservi@#1#2[#3]{%
1899   \glstrassignfieldfont{#2}%
1900   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1901   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glstentryuservi{#2}#3}}}%
1902 }
```

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

`\@acrshort` No case change.

```
1903 \def\@acrshort#1#2[#3]{%
1904   \glsdoifexists{#2}%
1905   {%
1906     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1907     \let\glstrifwasfirstuse\@secondoftwo
1908     \let\glsifplural\@secondoftwo
1909     \let\glscapscase\@firstofthree
1910     \let\glsinsert\@empty
1911     \def\glscustomtext{%
1912       \acronymfont{\glsaccessshort{#2}}#3%
1913     }%
1914     \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
1915   }%
1916   \glspostlinkhook
1917 }
```

`\@Acrshort` First letter uppercase.

```
1918 \def\@Acrshort#1#2[#3]{%
1919   \glsdoifexists{#2}%
1920   {%
1921     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1922     \let\glstrifwasfirstuse\@secondoftwo
```

```

1923 \let\glsifplural\@secondoftwo
1924 \let\glscapscase\@secondofthree
1925 \let\glsinsert\@empty
1926 \def\glscustomtext{%
1927 \acronymfont{\Glsaccessshort{#2}}#3%
1928 }%
1929 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1930 }%
1931 \glspostlinkhook
1932 }

```

\@ACRshort All uppercase.

```

1933 \def\@ACRshort#1#2[#3]{%
1934 \glsdoifexists{#2}%
1935 {%
1936 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1937 \let\glsxtrifwasfirstuse\@secondoftwo
1938 \let\glsifplural\@secondoftwo
1939 \let\glsapspace\@thirdofthree
1940 \let\glsinsert\@empty
1941 \def\glscustomtext{%
1942 \mfirstucMakeUppercase{\acronymfont{\glsaccessshort{#2}}#3}%
1943 }%
1944 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1945 }%
1946 \glspostlinkhook
1947 }

```

\@acrshortpl No case change.

```

1948 \def\@acrshortpl#1#2[#3]{%
1949 \glsdoifexists{#2}%
1950 {%
1951 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1952 \let\glsxtrifwasfirstuse\@secondoftwo
1953 \let\glsifplural\@firstoftwo
1954 \let\glsapspace\@firstofthree
1955 \let\glsinsert\@empty
1956 \def\glscustomtext{%
1957 \acronymfont{\glsaccessshortpl{#2}}#3%
1958 }%
1959 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1960 }%
1961 \glspostlinkhook
1962 }

```

\@Acrshortpl First letter uppercase.

```

1963 \def\@Acrshortpl#1#2[#3]{%
1964 \glsdoifexists{#2}%
1965 {%

```

```

1966 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1967 \let\glxtrifwasfirstuse\@secondoftwo
1968 \let\gl@sifplural\@firstoftwo
1969 \let\glscapscase\@secondofthree
1970 \let\gl@sinsert\@empty
1971 \def\glscustomtext{%
1972 \acronymfont{\Glsaccessshortpl{#2}}#3%
1973 }%
1974 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1975 }%
1976 \glspostlinkhook
1977 }

```

\@ACRshortpl All uppercase.

```

1978 \def\@ACRshortpl#1#2[#3]{%
1979 \glstoifexists{#2}%
1980 {%
1981 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1982 \let\glxtrifwasfirstuse\@secondoftwo
1983 \let\gl@sifplural\@firstoftwo
1984 \let\glscapscase\@thirdofthree
1985 \let\gl@sinsert\@empty
1986 \def\glscustomtext{%
1987 \mfirstucMakeUppercase{\acronymfont{\Glsaccessshortpl{#2}}#3}%
1988 }%
1989 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
1990 }%
1991 \glspostlinkhook
1992 }

```

\@acrlong No case change.

```

1993 \def\@acrlong#1#2[#3]{%
1994 \glstoifexists{#2}%
1995 {%
1996 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
1997 \let\glxtrifwasfirstuse\@secondoftwo
1998 \let\gl@sifplural\@secondoftwo
1999 \let\glscapscase\@firstofthree
2000 \let\gl@sinsert\@empty
2001 \def\glscustomtext{%
2002 \acronymfont{\Glsaccesslong{#2}}#3%
2003 }%
2004 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2005 }%
2006 \glspostlinkhook
2007 }

```

\@Acrlong First letter uppercase.

```

2008 \def\@Acrlong#1#2[#3]{%

```

```

2009 \glsdoifexists{#2}%
2010 {%
2011 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2012 \let\glxtrifwasfirstuse\@secondoftwo
2013 \let\glsifplural\@secondoftwo
2014 \let\glscapscase\@secondofthree
2015 \let\glsinsert\@empty
2016 \def\glscustomtext{%
2017 \acronymfont{\Glsaccesslong{#2}}#3%
2018 }%
2019 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2020 }%
2021 \glspostlinkhook
2022 }

```

\@ACRlong All uppercase.

```

2023 \def\@ACRlong#1#2[#3]{%
2024 \glsdoifexists{#2}%
2025 {%
2026 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2027 \let\glxtrifwasfirstuse\@secondoftwo
2028 \let\glsifplural\@secondoftwo
2029 \let\glsapsacase\@thirdofthree
2030 \let\glsinsert\@empty
2031 \def\glscustomtext{%
2032 \mfirstucMakeUppercase{\acronymfont{\glsaccesslong{#2}}#3}%
2033 }%
2034 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2035 }%
2036 \glspostlinkhook
2037 }

```

\@acrlongpl No case change.

```

2038 \def\@acrlongpl#1#2[#3]{%
2039 \glsdoifexists{#2}%
2040 {%
2041 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2042 \let\glxtrifwasfirstuse\@secondoftwo
2043 \let\glsifplural\@firstoftwo
2044 \let\glsapsacase\@firstofthree
2045 \let\glsinsert\@empty
2046 \def\glscustomtext{%
2047 \acronymfont{\glsaccesslongpl{#2}}#3%
2048 }%
2049 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2050 }%
2051 \glspostlinkhook
2052 }

```

`\@Acrlongpl` First letter uppercase.

```
2053 \def\@Acrlongpl#1#2[#3]{%
2054   \glsdoifexists{#2}%
2055   {%
2056     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2057     \let\glxtrifwasfirstuse\@secondoftwo
2058     \let\glsifplural\@firstoftwo
2059     \let\glsapscase\@secondofthree
2060     \let\glsinsert\@empty
2061     \def\glscustomtext{%
2062       \acronymfont{\Glsaccesslongpl{#2}}#3%
2063     }%
2064     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2065   }%
2066   \glspostlinkhook
2067 }
```

`\@ACRlongpl` All uppercase.

```
2068 \def\@ACRlongpl#1#2[#3]{%
2069   \glsdoifexists{#2}%
2070   {%
2071     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2072     \let\glxtrifwasfirstuse\@secondoftwo
2073     \let\glsifplural\@firstoftwo
2074     \let\glsapscase\@thirdofthree
2075     \let\glsinsert\@empty
2076     \def\glscustomtext{%
2077       \mfirstucMakeUppercase{\acronymfont{\Glsaccesslongpl{#2}}#3}%
2078     }%
2079     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2080   }%
2081   \glspostlinkhook
2082 }
```

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

`\@glsaddkey`

```
2083 \renewcommand*{\@glsaddkey}[7]{%
2084   \key@ifundefined{glossentry}{#1}%
2085   {%
2086     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
2087     \appto\@gls@keymap{,#1}{#1}}%
2088   \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
2089   \appto\@newglossaryentryposthook{%
2090     \letcs{\@glo@tmp}{@glo@#1}%
2091     \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
2092   }%
2093   \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
```



```
2094 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%
```

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

```
2095 \ifcsdef{@gls@user@#1@}%
2096 {%
2097 \PackageError{glossaries}%
2098 {Can't define '\string#5' as helper command
2099 '\expandafter\string\csname @gls@user@#1@\endcsname' already
2100 exists}%
2101 {}}%
2102 }%
2103 {%
2104 \expandafter\newcommand\expandafter*\expandafter
2105 {\csname @gls@user@#1@\endcsname}[2][ ]{%
2106 \new@ifnextchar[%
2107 {\csuse{@gls@user@#1@}{##1}{##2}}%
2108 {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
2109 \csdef{@gls@user@#1@}##1##2[##3]{%
2110 \@gls@field@link{##1}{##2}{#3{##2}##3}%
2111 }%
2112 \newrobustcmd*{#5}{%
2113 \expandafter\@gls@hyp@opt\csname @gls@user@#1@\endcsname}%
2114 }%
```

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

```
2115 \ifcsdef{@Gls@user@#1@}%
2116 {%
2117 \PackageError{glossaries}%
2118 {Can't define '\string#6' as helper command
2119 '\expandafter\string\csname @Gls@user@#1@\endcsname' already
2120 exists}%
2121 {}}%
2122 }%
2123 {%
2124 \expandafter\newcommand\expandafter*\expandafter
2125 {\csname @Gls@user@#1@\endcsname}[2][ ]{%
2126 \new@ifnextchar[%
2127 {\csuse{@Gls@user@#1@}{##1}{##2}}%
2128 {\csuse{@Gls@user@#1@}{##1}{##2}[ ]}}%
2129 \csdef{@Gls@user@#1@}##1##2[##3]{%
2130 \@gls@field@link[\let\gls@caps@case\@secondofthree]%
2131 {##1}{##2}{#4{##2}##3}%
2132 }%
2133 \newrobustcmd*{#6}{%
2134 \expandafter\@gls@hyp@opt\csname @Gls@user@#1@\endcsname}%
2135 }%
```

Finally the all caps version (modified):

```
2136 \ifcsdef{@GLS@user@#1@}%
2137 {%
2138 \PackageError{glossaries}%
```

```

2139     {Can't define '\string#7' as helper command
2140     '\expandafter\string\csname @GLS@user@#1@endcsname' already
2141     exists}%
2142   }%
2143 }%
2144 {%
2145   \expandafter\newcommand\expandafter*\expandafter
2146   {\csname @GLS@user@#1@endcsname}[2][ ]{%
2147     \new@ifnextchar[%
2148       {\csuse{@GLS@user@#1@}{##1}{##2}}%
2149       {\csuse{@GLS@user@#1@}{##1}{##2}[ ]}}%
2150   \csdef{@GLS@user@#1@}##1##2[##3]{%
2151     \@gls@field@link[\let\gls@scaps@case\@thirdofthree]%
2152     {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
2153   }%
2154   \newrobustcmd*{#7}{%
2155     \expandafter\@gls@hyp@opt\csname @GLS@user@#1@endcsname}%
2156   }%
2157 }%
2158 {%
2159   \PackageError{glossaries-extra}{Key '#1' already exists}{}%
2160 }%
2161 }

```

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

```
2162 \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.

```
2163 \let\@gls@xtr@org@checkfirsthyper\@gls@link@checkfirsthyper
```

```
2164 \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.

```
2165 \ifglsused{\glslabel}%
```

```
2166   {\let\glsxtrifwasfirstuse\@secondoftwo}
```

```
2167   {\let\glsxtrifwasfirstuse\@firstoftwo}%
```

Store the category label for convenience.

```
2168 \edef\gls@categorylabel{\gls@category{\glslabel}}%
```

```
2169 \ifglsused{\glslabel}%
```

```
2170 {%
```

```
2171   \glsifcategoryattribute{\gls@categorylabel}{nohypernext}{true}%
```

```
2172     {\KV@gls@link@hyperfalse}}%
```

```
2173 }%
```

```
2174 {%
```

```
2175   \glsifcategoryattribute{\gls@categorylabel}{nohyperfirst}{true}%
```

```

2176     {\KV@glslink@hyperfalse}{}%
2177 }%
2178 \glslinkcheckfirsthyperhook
2179 }

```

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.

```

2180 \ifdef\do@gl:disablehyperinlist
2181 {%
2182   \let\@gl:extr@do@gl:disablehyperinlist\do@gl:disablehyperinlist
2183   \renewcommand*{\do@gl:disablehyperinlist}{%
2184     \@gl:extr@do@gl:disablehyperinlist
2185     \gl:sifattribute{\gl:label}{nohyper}{true}{\KV@glslink@hyperfalse}{}%
2186   }
2187 }
2188 {}

```

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

```

2189 \define@boolkey{gl:link}{noindex}[true]{}
2190 \KV@glslink@noindexfalse

```

If `\@gl:issetdefault@gl:link@opts` has been defined (glossaries v4.20) use it to set the default keys in `\@gl:link`.

lt@gl:link@opts

```

2191 \ifdef\@gl:issetdefault@gl:link@opts
2192 {
2193   \renewcommand*{\@gl:issetdefault@gl:link@opts}{%
2194     \KV@glslink@noindexfalse
2195     \@gl:extr:setaliasnoindex
2196   }
2197 }
2198 {

```

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

```

2199   \newcommand*{\@gl:issetdefault@gl:link@opts}{%
2200     \KV@glslink@noindexfalse
2201     \@gl:extr:setaliasnoindex
2202   }
2203   \preto\do@gl:disablehyperinlist{\@gl:issetdefault@gl:link@opts}
2204 }

```

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.)

```

2205 \providecommand*\gl:extr:setaliasnoindex{}%
2206 \KV@glslink@noindextrue
2207 }

```

setaliasnoindex

```
2208 \newcommand*{\@glxtrsetaliasnoindex}{%
2209 \ifglshasfield{alias}{\glslabel}%
2210 {%
2211 \let\glxtrindexaliased\@glxtrindexaliased
2212 \glxtrsetaliasnoindex
2213 \let\glxtrindexaliased\@no@glxtrindexaliased
2214 }%
2215 {}}%
2216 }
```

xtrindexaliased

```
2217 \newcommand{\@glxtrindexaliased}{%
2218 \ifKV@glslink@noindex
2219 \else
2220 \begingroup
2221 \let\@glsnnumberformat\@glxtr@defaultnumberformat
2222 \edef\@glscounter{\csname glo@glsdetoklabel{\glslabel}@counter\endcsname}%
2223 \glxtr@saveentrycounter
2224 \@do@wrglossary{\glxtralias{\glslabel}}%
2225 \endgroup
2226 \fi
2227 }
```

xtrindexaliased

```
2228 \newcommand{\@no@glxtrindexaliased}{%
2229 \PackageError{glossaries-extra}{\string\glxtrindexaliased\space
2230 not permitted outside definition of \string\glxtrsetaliasnoindex}%
2231 {}}%
2232 }
```

xtrindexaliased Provide a command to redirect alias indexing, but only allow it to be used within \glxtrsetaliasnoindex.

```
2233 \let\glxtrindexaliased\@no@glxtrindexaliased
```

tDefaultGlsOpts Set the default options for \glslink etc.

```
2234 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
2235 \renewcommand*{\@glsc@setdefault@glslink@opts}{%
2236 \setkeys{glslink}{#1}%
2237 \@glxtrsetaliasnoindex
2238 }%
2239 }
```

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

```
2240 \newcommand*{\glxtrifindexing}[2]{%
2241 \ifKV@glslink@noindex #2\else #1\fi
2242 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```

2243 \renewcommand*{\glswriteentry}[2]{%
2244   \glsxtrifindexing
2245   {%
2246     \ifglsindexonlyfirst
2247     \ifglsused{#1}
2248     {\glsxtrdoautoindexname{#1}{dualindex}}%
2249     {#2}%
2250   \else
2251     \glsifattribute{#1}{indexonlyfirst}{true}%
2252     {\ifglsused{#1}
2253     {\glsxtrdoautoindexname{#1}{dualindex}}%
2254     {#2}}%
2255     {#2}%
2256   \fi
2257 }%
2258 {}%
2259 }

```

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

```

2260 \appto\@do@wrglossary{\@glsxtr@do@wrindex
2261   \glsxtrdowrglossaryhook{\@gls@label}%
2262 }

```

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

Similarly for the “noidx” version:

`s@noidxglossary`

```

2263 \appto\gls@noidxglossary{\@glsxtr@do@wrindex
2264   \glsxtrdowrglossaryhook{\@gls@label}%
2265 }

```

`xtr@do@wrindex`

```

2266 \newcommand*{\@glsxtr@do@wrindex}{%
2267   \glsxtrdoautoindexname{\@gls@label}{dualindex}%
2268 }

```

`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.)

```

2269 \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.

```

2270 \newcommand*{\@gls@alt@hyp@opt}[1]{%
2271   \let\glslinkvar\@firstofthree
2272   \let\@gls@hyp@opt@cs#1\relax
2273   \@ifstar{\s@gls@hyp@opt}%
2274   {\@ifnextchar+%

```

```

2275 {\@firstoftwo{\p@glshyp@opt}}%
2276 {%
2277 \expandafter\@ifnextchar\@glshalt@hyp@opt@char
2278 {\@firstoftwo{\@alt@glshyp@opt}}%
2279 {#1}%
2280 }%
2281 }%
2282 }

```

`\alt@glshyp@opt` User version

```

2283 \newcommand*\@alt@glshyp@opt}[1][{}]{%
2284 \let\glslinkvar\@firstofthree
2285 \expandafter\@glshyp@opt@cs\expandafter[\@glshalt@hyp@opt@keys,#1]}

```

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

```

2286 \newcommand*\@glshalt@hyp@opt@char{}

```

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

```

2287 \newcommand*\@glshalt@hyp@opt@keys{}

```

`\rSetAltModifier`

```

2288 \newcommand*\GlsXtrSetAltModifier}[2]{%
2289 \let\@glshyp@opt\@glshalt@hyp@opt
2290 \def\@glshalt@hyp@opt@char{#1}%
2291 \def\@glshalt@hyp@opt@keys{#2}%
2292 }

```

`\glsdohyperlink` Unpleasant complications can occur if the text or first key etc contains `\glshyp`, particularly if there are hyperlinks. To get around this problem, patch `\glsdohyperlink` so that it temporarily makes `\glshyp` behave like `\glstext` [*hyper=false,noindex*]. (This will be overridden if the user explicitly cancels either of those options in the optional argument of `\glshyp` or using the plus version.) This also patches the short form commands like `\acrshort` and `\glsxtrshort` to use `\glstentryshort` and, similarly, the long form commands like `\acrlong` and `\glsxtrlong` to use `\glstentrylong`. Added attribute check.

```

2293 \renewcommand*\glsdohyperlink}[2]{%
2294 \glshasattribute{\glslabel}{targeturl}%
2295 {%
2296 \glshasattribute{\glslabel}{targetname}%
2297 {%
2298 \glshasattribute{\glslabel}{targetcategory}%
2299 {%
2300 \hyperref{\glsggetattribute{\glslabel}{targeturl}}%
2301 {\glsggetattribute{\glslabel}{targetcategory}}%
2302 {\glsggetattribute{\glslabel}{targetname}}%
2303 {\glsxtrprotectlinks#2}}%
2304 }%
2305 }%
2306 \hyperref{\glsggetattribute{\glslabel}{targeturl}}%

```

```

2307     {}%
2308     {\glsgetattribute{\glslabel}{targetname}}%
2309     {\glsxtrprotectlinks#2}}%
2310   }%
2311 }%
2312 {%
2313   \href{\glsgetattribute{\glslabel}{targeturl}}%
2314     {\glsxtrprotectlinks#2}}%
2315 }%
2316 }%
2317 {%

```

Check for alias.

```

2318   \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
2319   \ifdefvoid\gloaliaslabel
2320   {%
2321     \glsxtrhyperlink{#1}{\glsxtrprotectlinks#2}}%
2322   }%
2323   {%

```

Redirect link to the alias target.

```

2324   \glsxtrhyperlink
2325   {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
2326   {\glsxtrprotectlinks#2}}%
2327   }%
2328 }%
2329 }

```

`\glsxtrhyperlink` Allows integration with the base glossaries package's `debug=showtargets` option.

```

2330 \ifdef\glsshowtarget
2331 {
2332   \newcommand{\glsxtrhyperlink}[2]{%
2333     \@glsshowtarget{#1}%
2334     \hyperlink{#1}{#2}}%
2335   }%
2336 }
2337 {
2338   \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%
2339 }

```

`\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.

```

2340 \renewrobustcmd*{\glsxtrhyperlink}[2][\glsentrytext{\@glo@label}]{%
2341   \def\@glo@label{#2}%
2342   {\edef\glslabel{#2}}%
2343   \@glslink{\glolinkprefix\glslabel}{#1}}%
2344 }

```

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

```
2345 \ifundef\glsdonohyperlink
2346 {%
2347   \renewcommand{\glsdisablehyper}{%
2348     \KV@glslink@hyperfalse
2349     \let\@glslink\glsdonohyperlink
2350     \let\@glstarget\@secondoftwo
2351   }
2352 }
2353 }
```

`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.

```
2354 \def\glsdonohyperlink#1#2{\@glsxtrprotectlinks #2}
```

Reset `\@glslink` with patched versions:

```
2355 \ifcsundef{hyperlink}%
2356 {%
2357   \let\@glslink\glsdonohyperlink
2358 }%
2359 {%
2360   \let\@glslink\glsdohyperlink
2361 }
```

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

```
2362 \newcommand*{\glsxtrprotectlinks}{%
2363   \KV@glslink@hyperfalse
2364   \KV@glslink@noindextrue
2365   \let\@gls@\@glsxtr@p@text@
2366   \let\@Gls@\@Glsxtr@p@text@
2367   \let\@GLS@\@GLSxtr@p@text@
2368   \let\@glspl@\@glsxtr@p@plural@
2369   \let\@Glspl@\@Glsxtr@p@plural@
2370   \let\@GLSpl@\@GLSxtr@p@plural@
2371   \let\@glsxtrshort@\@glsxtr@p@short@
2372   \let\@Glsxtrshort@\@Glsxtr@p@short@
2373   \let\@GLSxtrshort@\@GLSxtr@p@short@
2374   \let\@glsxtrlong@\@glsxtr@p@long@
2375   \let\@Glsxtrlong@\@Glsxtr@p@long@
2376   \let\@GLSxtrlong@\@GLSxtr@p@long@
2377   \let\@glsxtrshortpl@\@glsxtr@p@shortpl@
2378   \let\@Glsxtrshortpl@\@Glsxtr@p@shortpl@
2379   \let\@GLSxtrshortpl@\@GLSxtr@p@shortpl@
2380   \let\@glsxtrlongpl@\@glsxtr@p@longpl@
2381   \let\@Glsxtrlongpl@\@Glsxtr@p@longpl@
2382   \let\@GLSxtrlongpl@\@GLSxtr@p@longpl@
```



```

2383 \let\@acrshort\@glsxtr@p@acrshort@
2384 \let\@Acrshort\@Glsxtr@p@acrshort@
2385 \let\@ACRshort\@GLSxtr@p@acrshort@
2386 \let\@acrshortpl\@glsxtr@p@acrshortpl@
2387 \let\@Acrshortpl\@Glsxtr@p@acrshortpl@
2388 \let\@ACRshortpl\@GLSxtr@p@acrshortpl@
2389 \let\@acrlong\@glsxtr@p@acrlong@
2390 \let\@Acrlong\@Glsxtr@p@acrlong@
2391 \let\@ACRlong\@GLSxtr@p@acrlong@
2392 \let\@acrlongpl\@glsxtr@p@acrlongpl@
2393 \let\@Acrlongpl\@Glsxtr@p@acrlongpl@
2394 \let\@ACRlongpl\@GLSxtr@p@acrlongpl@
2395 }

```

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

@glsxtr@p@text@

```
2396 \def\@glsxtr@p@text@#1#2[#3]{\@gls@text@{#1}{#2}[#3]}
```

@Glsxtr@p@text@

```
2397 \def\@Glsxtr@p@text@#1#2[#3]{\@Gls@text@{#1}{#2}[#3]}
```

@GLSxtr@p@text@

```
2398 \def\@GLSxtr@p@text@#1#2[#3]{\@GLS@text@{#1}{#2}[#3]}
```

lsxtr@p@plural@

```
2399 \def\@glsxtr@p@plural@#1#2[#3]{\@gls@plural@{#1}{#2}[#3]}
```

Glsxtr@p@plural@

```
2400 \def\@Glsxtr@p@plural@#1#2[#3]{\@Gls@plural@{#1}{#2}[#3]}
```

LSxtr@p@plural@

```
2401 \def\@GLSxtr@p@plural@#1#2[#3]{\@GLS@plural@{#1}{#2}[#3]}
```

glsxtr@p@short@

```

2402 \def\@glsxtr@p@short@#1#2[#3]{%
2403   {%
2404     \glssetabbrvfmt{\gls@category{#2}}%
2405     \glsabbrvfont{\gls@entryshort{#2}}#3%
2406   }%
2407 }

```

Glsxtr@p@short@

```

2408 \def\@Glsxtr@p@short@#1#2[#3]{%
2409   {%
2410     \glssetabbrvfmt{\Gls@category{#2}}%
2411     \glsabbrvfont{\Gls@entryshort{#2}}#3%
2412   }%
2413 }

```

GLSxtr@p@short@

```
2414 \def\@GLSxtr@p@short@#1#2[#3]{%
2415   {%
2416     \glsssetabbrvfmt{\glscategory{#2}}%
2417     \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshort{#2}}#3}%
2418   }%
2419 }
```

sxtr@p@shortpl@

```
2420 \def\@glsxtr@p@shortpl@#1#2[#3]{%
2421   {%
2422     \glsssetabbrvfmt{\glscategory{#2}}%
2423     \glsabbrvfont{\glsentryshortpl{#2}}#3%
2424   }%
2425 }
```

Sxtr@p@shortpl@

```
2426 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2427   {%
2428     \glsssetabbrvfmt{\glscategory{#2}}%
2429     \glsabbrvfont{\Glsentryshortpl{#2}}#3%
2430   }%
2431 }
```

Sxtr@p@shortpl@

```
2432 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2433   {%
2434     \glsssetabbrvfmt{\glscategory{#2}}%
2435     \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshortpl{#2}}#3}%
2436   }%
2437 }
```

@glsxtr@p@long@

```
2438 \def\@glsxtr@p@long@#1#2[#3]{\glsentrylong{#2}#3}
```

@Glsxtr@p@long@

```
2439 \def\@Glsxtr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}
```

@GLSxtr@p@long@

```
2440 \def\@GLSxtr@p@long@#1#2[#3]{%
2441   {\mfirstucMakeUppercase{\glslongfont{\glsentrylong{#2}}#3}}
```

lsxtr@p@longpl@

```
2442 \def\@glsxtr@p@longpl@#1#2[#3]{\glsentrylongpl{#2}#3}
```

lSxtr@p@longpl@

```
2443 \def\@Glsxtr@p@longpl@#1#2[#3]{\glslongfont{\Glsentrylongpl{#2}}#3}
```

LSxtr@p@longpl@

```
2444 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2445   {\mfirstucMakeUppercase{\glslongfont{\glsentrylongpl{#2}}#3}}}
```

xtr@p@acrshort@

```
2446 \def\@glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\glsentryshort{#2}}#3}}
```

xtr@p@acrshort@

```
2447 \def\@GLSxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}}
```

xtr@p@acrshort@

```
2448 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2449   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}
```

r@p@acrshortpl@

```
2450 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\glsentryshortpl{#2}}#3}}
```

r@p@acrshortpl@

```
2451 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}}
```

r@p@acrshortpl@

```
2452 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2453   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}
```

sxtr@p@acrlong@

```
2454 \def\@glsxtr@p@acrlong@#1#2[#3]{\glsentrylong{#2}#3}}
```

sxtr@p@acrlong@

```
2455 \def\@GLSxtr@p@acrlong@#1#2[#3]{\Glsentrylong{#2}#3}}
```

Sxtr@p@acrlong@

```
2456 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2457   {\mfirstucMakeUppercase{\glsentrylong{#2}#3}}}
```

tr@p@acrlongpl@

```
2458 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\glsentrylongpl{#2}#3}}
```

tr@p@acrlongpl@

```
2459 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{\Glsentrylongpl{#2}#3}}
```

tr@p@acrlongpl@

```
2460 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2461   {\mfirstucMakeUppercase{\glsentrylongpl{#2}#3}}}
```

Commands to minimise conflict.

\@glsxtrp@opt

```
2462 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}
```

`\glsxtrsetpopts` Used in glossary to switch hyperlinks on for the `\glsxtrp` type of commands.

```
2463 \newcommand*{\glsxtrsetpopts}[1]{%
2464   \renewcommand*{\@glsxtrp@opt}{#1}%
2465 }
```

`\glossxtrsetpopts` Used in glossary to switch hyperlinks on for the `\glsxtrp` type of commands.

```
2466 \newcommand*{\glossxtrsetpopts}{%
2467   \glsxtrsetpopts{noindex}%
2468 }
```

`\@glsxtrp`

```
2469 \newrobustcmd*{\@glsxtrp}[2]{%
  Add scope.
2470   {%
2471     \let\glspostlinkhook\relax
2472     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
2473   }%
2474 }
```

`\@glsxtrp`

```
2475 \newrobustcmd*{\@glsxtrp}[2]{%
2476   \ifcsdef{gls#1}%
2477     {%
2478       \@glsxtrp{gls#1}{#2}%
2479     }%
2480   {%
2481     \ifcsdef{glsxtr#1}%
2482       {%
2483         \@glsxtrp{glsxtr#1}{#2}%
2484       }%
2485     {%
2486       \PackageError{glossaries-extra}{‘#1’ not recognised by
2487         \string\glsxtrp}{}%
2488     }%
2489   }%
2490 }
```

`\@Glsxtrp`

```
2491 \newrobustcmd*{\@Glsxtrp}[2]{%
2492   \ifcsdef{Gls#1}%
2493     {%
2494       \@glsxtrp{Gls#1}{#2}%
2495     }%
2496   {%
2497     \ifcsdef{Glsxtr#1}%
2498       {%
2499         \@glsxtrp{Glsxtr#1}{#2}%
2500       }%
2501     }
```

```

2501   {%
2502     \PackageError{glossaries-extra}{‘#1’ not recognised by
2503       \string\Glsxtrp}{}%
2504   }%
2505 }%
2506 }

```

`\@GLSxtrp`

```

2507 \newrobustcmd*{\@GLSxtrp}[2]{%
2508   \ifcsdef{GLS#1}%
2509   {%
2510     \@glsxtrp{GLS#1}{#2}%
2511   }%
2512   {%
2513     \ifcsdef{GLSxtr#1}%
2514     {%
2515       \@glsxtrp{GLSxtr#1}{#2}%
2516     }%
2517     {%
2518       \PackageError{glossaries-extra}{‘#1’ not recognised by
2519         \string\GLSxtrp}{}%
2520     }%
2521   }%
2522 }

```

`\glsxtr@entry@p`

```

2523 \newrobustcmd*{\glsxtr@headentry@p}[2]{%
2524   \glsifattribute{#1}{headuc}{true}%
2525   {%
2526     \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
2527   }%
2528   {%
2529     \@gls@entry@field{#1}{#2}%
2530   }%
2531 }

```

`\glsxtrp` Not robust as it needs to expand somewhat.

```

2532 \ifdef\texorpdfstring
2533 {
2534   \newcommand{\glsxtrp}[2]{%
2535     \protect\NoCaseChange
2536     {%
2537       \protect\texorpdfstring
2538       {%
2539         \protect\glsxtrifinmark
2540         {%
2541           \ifcsdef{glsxtrhead#1}%
2542           {%
2543             {\protect\csuse{glsxtrhead#1}{#2}}%

```

```

2544     }%
2545     {%
2546         \glxtr@headentry@p{#2}{#1}%
2547     }%
2548 }%
2549     {%
2550         \@glxtrp{#1}{#2}%
2551     }%
2552 }%
2553     {%
2554         \protect\@gls@entry@field{#2}{#1}%
2555     }%
2556 }%
2557 }
2558 }
2559 {
2560 \newcommand{\glxtrp}[2]{%
2561     \protect\NoCaseChange
2562     {%
2563         \protect\glxtrifinmark
2564         {%
2565             \ifcsdef{glxtrhead#1}%
2566             {%
2567                 {\protect\csuse{glxtrhead#1}}%
2568             }%
2569             {%
2570                 \glxtr@headentry@p{#2}{#1}%
2571             }%
2572         }%
2573         {%
2574             \@glxtrp{#1}{#2}%
2575         }%
2576     }%
2577 }
2578 }

```

Provide short synonyms for the most common option.

`\glsps`

```
2579 \newcommand*{\glsps}{\glxtrp{short}}
```

`\glspt`

```
2580 \newcommand*{\glspt}{\glxtrp{text}}
```

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

```
2581 \ifdef\teorpdfstring
```

```
2582 {
```

```
2583   \newcommand{\Glsxtrp}[2]{%
```

```

2584 \protect\NoCaseChange
2585 {%
2586 \protect\texorpdfstring
2587 {%
2588 \protect\glxtrifinmark
2589 {%
2590 \ifcsdef{Glsxtrhead#1}%
2591 {%
2592 {\protect\csuse{Glsxtrhead#1}{#2}}%
2593 }%
2594 {%
2595 \protect\@Gls@entry@field{#2}{#1}%
2596 }%
2597 }%
2598 {%
2599 \@Glsxtrp{#1}{#2}%
2600 }%
2601 }%
2602 {%
2603 \protect\@gls@entry@field{#2}{#1}%
2604 }%
2605 }%
2606 }
2607 }
2608 {
2609 \newcommand{\Glsxtrp}[2]{%
2610 \protect\NoCaseChange
2611 {%
2612 \protect\glxtrifinmark
2613 {%
2614 \ifcsdef{Glsxtrhead#1}%
2615 {%
2616 {\protect\csuse{Glsxtrhead#1}}%
2617 }%
2618 {%
2619 \protect\@Gls@entry@field{#2}{#1}%
2620 }%
2621 }%
2622 {%
2623 \@Glsxtrp{#1}{#2}%
2624 }%
2625 }%
2626 }
2627 }

```

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

```

2628 \ifdef\texorpdfstring
2629 {
2630 \newcommand{\GLSxtrp}[2]{%

```

```

2631 \protect\NoCaseChange
2632 {%
2633 \protect\texorpdfstring
2634 {%
2635 \protect\glstrifinmark
2636 {%
2637 \ifcsdef{GLSxtr#1}%
2638 {%
2639 {\protect\GLSxtrshort [noindex,hyper=false]{#1} []}%
2640 }%
2641 {%
2642 \protect\mfirstucMakeUppercase
2643 {%
2644 \protect\@gls@entry@field{#2}{#1}%
2645 }%
2646 }%
2647 }%
2648 {%
2649 \@GLSxtrp{#1}{#2}%
2650 }%
2651 }%
2652 {%
2653 \protect\@gls@entry@field{#2}{#1}%
2654 }%
2655 }%
2656 }
2657 }
2658 {
2659 \newcommand{\GLSxtrp}[2]{%
2660 \protect\NoCaseChange
2661 {%
2662 \protect\glstrifinmark
2663 {%
2664 \ifcsdef{GLSxtr#1}%
2665 {%
2666 {\protect\GLSxtrshort [noindex,hyper=false]{#1} []}%
2667 }%
2668 {%
2669 \protect\mfirstucMakeUppercase
2670 {%
2671 \protect\@gls@entry@field{#2}{#1}%
2672 }%
2673 }%
2674 }%
2675 {%
2676 \@GLSxtrp{#1}{#2}%
2677 }%
2678 }%
2679 }

```


2680 }

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.

```
2681 \renewcommand*\@glsunset}[1]{%
2682   \@glsunset{#1}%
2683   \glsxtrpostunset{#1}%
2684 }%
```

glsxtrpostunset

```
2685 \newcommand*\glsxtrpostunset}[1]{}
```

\@glslocalunset Local unset.

```
2686 \renewcommand*\@glslocalunset}[1]{%
2687   \@glslocalunset{#1}%
2688   \glsxtrpostlocalunset{#1}%
2689 }%
```

rpostlocalunset

```
2690 \newcommand*\glsxtrpostlocalunset}[1]{}
```

\@glsreset Global reset.

```
2691 \renewcommand*\@glsreset}[1]{%
2692   \@glsreset{#1}%
2693   \glsxtrpostreset{#1}%
2694 }%
```

glsxtrpostreset

```
2695 \newcommand*\glsxtrpostreset}[1]{}
```

\@glslocalreset Local reset.

```
2696 \renewcommand*\@glslocalreset}[1]{%
2697   \@glslocalreset{#1}%
2698   \glsxtrpostlocalreset{#1}%
2699 }%
```

rpostlocalreset

```
2700 \newcommand*\glsxtrpostlocalreset}[1]{}
```

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

```
2701 \newcommand*\GlsXtrEnableEntryCounting}[2]{%
```

Enable entry counting:

```
2702 \glsenableentrycount
```

Redefine \gls etc:

```
2703 \renewcommand*{\gls}{\cglsl}%
2704 \renewcommand*{\Gls}{\cGls}%
2705 \renewcommand*{\glspl}{\cglspl}%
2706 \renewcommand*{\Glspl}{\cGlspl}%
2707 \renewcommand*{\GLS}{\cGLS}%
2708 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2709 \@glsxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
2710 \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
2711 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
2712 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2713 can't be used with \string\GlsXtrEnableEntryCounting}%
2714 {Use one or other but not both commands}}%
2715 }
```

countunsetattr

```
2716 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
2717 \@for\@glsxtr@cat:=#1\do
2718 {%
2719 \ifdefempty{\@glsxtr@cat}{}%
2720 {%
2721 \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
2722 }%
2723 }%
2724 }
```

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

enableentrycount

```
2725 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
2726 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
```

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

```
2727 \renewcommand*{\gls@defdocnewglossaryentry}{%
2728 \renewcommand*\newglossaryentry[2]{%
2729 \PackageError{glossaries}{\string\newglossaryentry\space
2730 may only be used in the preamble when entry counting has
2731 been activated}{If you use \string\glsenableentrycount\space
2732 you must place all entry definitions in the preamble not in
2733 the document environment}%
2734 }%
2735 }%
```

New commands to access new fields:

```
2736 \newcommand*\glsentrycurrcount}[1]{%
2737 \ifcsundef{glo@glsdetoklabel{##1}@currcount}%
2738 {0}\@gls@entry@field{##1}{currcount}}%
2739 }%
2740 \newcommand*\glsentryprevcount}[1]{%
2741 \ifcsundef{glo@glsdetoklabel{##1}@prevcount}%
2742 {0}\@gls@entry@field{##1}{prevcount}}%
2743 }%
```

Adjust post unset and reset:

```
2744 \let@glsxtr@entrycount@org@unset@glsxtrpostunset
2745 \renewcommand*\glsxtrpostunset}[1]{%
2746 \@glsxtr@entrycount@org@unset{##1}%
2747 \@gls@increment@currcount{##1}%
2748 }%
2749 \let@glsxtr@entrycount@org@localunset@glsxtrpostlocalunset
2750 \renewcommand*\glsxtrpostlocalunset}[1]{%
2751 \@glsxtr@entrycount@org@localunset{##1}%
2752 \@gls@local@increment@currcount{##1}%
2753 }%
2754 \let@glsxtr@entrycount@org@reset@glsxtrpostreset
2755 \renewcommand*\glsxtrpostreset}[1]{%
2756 \@glsxtr@entrycount@org@reset{##1}%
2757 \csgdef{glo@glsdetoklabel{##1}@currcount}{0}%
2758 }%
2759 \let@glsxtr@entrycount@org@localreset@glsxtrpostlocalreset
2760 \renewcommand*\glsxtrpostlocalreset}[1]{%
2761 \@glsxtr@entrycount@org@localreset{##1}%
2762 \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
2763 }%
```

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

```
2764 \let@cGls@\@cGls@
2765 \let@cGlspl@\@cGlspl@

2766 \let@cGls@\@cGls@
2767 \let@cGlspl@\@cGlspl@
2768 \let@cGLS@\@cGLS@
2769 \let@cGLSpl@\@cGLSpl@
```

The rest is as the original definition.

```
2770 \AtEndDocument{\@gls@write@entrycounts}%
2771 \renewcommand*\@gls@entry@count}[2]{%
2772 \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
2773 }%
2774 \let@glsenableentrycount\relax
2775 \renewcommand*\glsenableentryunitcount}{%
2776 \PackageError{glossaries-extra}{\string@glsenableentryunitcount\space
```

```

2777     can't be used with \string\glsenableentrycount}%
2778     {Use one or other but not both commands}%
2779 }%
2780 }

```

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

```

2781 \renewcommand*{\@gls@write@entrycounts}{%
2782   \immediate\write\@auxout
2783   {\string\providecommand*\string\@gls@entry@count}[2]{}%
2784   \count@=0\relax
2785   \forallglsentries{\@glsentry}{%
2786     \glshasattribute{\@glsentry}{entrycount}%
2787     {%
2788       \ifglsused{\@glsentry}%
2789       {%
2790         \immediate\write\@auxout
2791         {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
2792       }%
2793     }%
2794     \advance\count@ by \@ne
2795   }%
2796 }%
2797 }%
2798 \ifnum\count@=0
2799   \GlossariesExtraWarningNoLine{Entry counting has been enabled
2800   \MessageBreak with \string\glsenableentrycount\space but the
2801   \MessageBreak attribute 'entrycount' hasn't
2802   \MessageBreak been assigned to any of the defined
2803   \MessageBreak entries}%
2804 \fi
2805 }

```

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

```

2806 \newcommand*\glsextrifcounttrigger[3]{%
2807   \glshasattribute{#1}{entrycount}%
2808   {%
2809     \ifnum\glsentryprevcount{#1}>\glsgetattribute{#1}{entrycount}\relax
2810     #3%
2811   \else
2812     #2%
2813   \fi
2814 }%
2815 {#3}%
2816 }

```

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

\@@cgl_s@

```
2817 \def\@@cgl_s@#1#2[#3]{%
2818   \glxtrifcounttrigger{#2}%
2819   {%
2820     \cgl_sformat{#2}{#3}%
2821     \gl_sunset{#2}%
2822   }%
2823   {%
2824     \@gl_s@{#1}{#2}[#3]%
2825   }%
2826 }%
```

\@@cglSpl@

```
2827 \def\@@cglSpl@#1#2[#3]{%
2828   \glxtrifcounttrigger{#2}%
2829   {%
2830     \cglSplformat{#2}{#3}%
2831     \gl_sunset{#2}%
2832   }%
2833   {%
2834     \@glSpl@{#1}{#2}[#3]%
2835   }%
2836 }%
```

\@@cGl_s@

```
2837 \def\@@cGl_s@#1#2[#3]{%
2838   \glxtrifcounttrigger{#2}%
2839   {%
2840     \cGl_sformat{#2}{#3}%
2841     \gl_sunset{#2}%
2842   }%
2843   {%
2844     \@Gl_s@{#1}{#2}[#3]%
2845   }%
2846 }%
```

\@@cGlSpl@

```
2847 \def\@@cGlSpl@#1#2[#3]{%
2848   \glxtrifcounttrigger{#2}%
2849   {%
2850     \cGlSplformat{#2}{#3}%
2851     \gl_sunset{#2}%
2852   }%
2853   {%
2854     \@GlSpl@{#1}{#2}[#3]%
2855   }%
2856 }%
```

\@@cGLS@

```
2857 \def\@@cGLS@#1#2[#3]{%
2858   \glxtrifcounttrigger{#2}%
2859   {%
2860     \cGLSformat{#2}{#3}%
2861     \glset{#2}%
2862   }%
2863   {%
2864     \@GLS@{#1}{#2}[#3]%
2865   }%
2866 }%
```

\@@cGLSp1@

```
2867 \def\@@cGLSp1@#1#2[#3]{%
2868   \glxtrifcounttrigger{#2}%
2869   {%
2870     \cGLSp1format{#2}{#3}%
2871     \glset{#2}%
2872   }%
2873   {%
2874     \@GLSp1@{#1}{#2}[#3]%
2875   }%
2876 }%
```

Remove default warnings from \cglis etc so that it can be used interchangeable with \glis etc.

\@cglis@

```
2877 \def\@cglis@#1#2[#3]{\@glis@{#1}{#2}[#3]}
```

\@cGls@

```
2878 \def\@cGls@#1#2[#3]{\@Gls@{#1}{#2}[#3]}
```

\@cglsp1@

```
2879 \def\@cglsp1@#1#2[#3]{\@glsp1@{#1}{#2}[#3]}
```

\@cGlspl@

```
2880 \def\@cGlspl@#1#2[#3]{\@Glspl@{#1}{#2}[#3]}
```

Add all upper case versions not provided by glossaries.

\cGLS

```
2881 \newrobustcmd*{\cGLS}{\@glshyp@opt\@cGLS}
```

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

```
2882 \newcommand*{\@cGLS}[2][ ]{%
2883   \new@ifnextchar[{\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2}[]}%
2884 }
```

\@cGLS@

```
2885 \def\@cGLS@#1#2[#3]{\@GLS@{#1}{#2}[#3]}
```

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

```
2886 \newcommand*\cGLSformat}[2]{%
2887 \expandafter\mfirstucMakeUppercase\expandafter{\cglformat{#1}{#2}}%
2888 }
```

\cGLSp1

```
2889 \newrobustcmd*\cGLSp1{\@gls@hyp@opt\@cGLSp1}
```

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

```
2890 \newcommand*\@cGLSp1}[2][ ]{%
2891 \new@ifnextchar[{\@cGLSp1@{#1}{#2}}{\@cGLSp1@{#1}{#2}[ ]}]%
2892 }
```

\@cGLSp1@

```
2893 \def\@cGLSp1@#1#2[#3]{\@GLSp1@{#1}{#2}[#3]}
```

\cGLSplformat Format used by \cGLSp1 if entry only used once on previous run. The first argument is the label, the second argument is the insert text.

```
2894 \newcommand*\cGLSplformat}[2]{%
2895 \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
2896 }
```

Modify the trigger formats to check for the regular attribute.

\cglformat

```
2897 \renewcommand*\cglformat}[2]{%
2898 \glsifregular{#1}
2899 {\glsentryfirst{#1}}%
2900 {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}#2%
2901 }
```

\cGlsformat

```
2902 \renewcommand*\cGlsformat}[2]{%
2903 \glsifregular{#1}
2904 {\Glsentryfirst{#1}}%
2905 {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}#2%
2906 }
```

\cglsplformat

```
2907 \renewcommand*\cglsplformat}[2]{%
2908 \glsifregular{#1}
2909 {\glsentryfirstplural{#1}}%
2910 {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}#2%
2911 }
```

\cGlsplformat

```
2912 \renewcommand*\cGlsplformat}[2]{%
2913   \glsifregular{#1}
2914   {\Glsentryfirstplural{#1}}%
2915   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2}%
2916 }
```

New code similar to above for unit counting.

defunitcounters

```
2917 \newcommand*\@@newglossaryentry@defunitcounters}{%
2918   \edef\@glo@countunit{\csuse{@glsxtr@categoryattr@\@glo@category @unitcount}}%
2919   \ifdefvoid\@glo@countunit
2920   {}%
2921   {%
2922     \@glsxtr@ifunitcounter{\@glo@countunit}%
2923     }%
2924     {\expandafter\@glsxtr@addunitcounter\expandafter{\@glo@countunit}}%
2925   }%
2926 }
```

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

```
2927 \newcommand*\@glsxtr@unitcountlist}{}
```

@addunitcounter

```
2928 \newcommand*\@glsxtr@addunitcounter}[1]{%
2929   \listadd{\@glsxtr@unitcountlist}{#1}%
2930   \ifcsundef{glsxtr@theunit@#1}
2931   {%
2932     \ifcsdef{theH#1}%
2933     {\csdef{glsxtr@theunit@#1}{\csuse{theH#1}}}%
2934     {\csdef{glsxtr@theunit@#1}{\csuse{the#1}}}%
2935   }%
2936   {}%
2937 }
```

r@ifunitcounter

```
2938 \newcommand*\@glsxtr@ifunitcounter}[3]{%
2939   \xifinlist{#1}{\@glsxtr@unitcountlist}{#2}{#3}%
2940 }
```

urrentunitcount

```
2941 \newcommand*\@glsxtr@currentunitcount[1]{%
2942   glo@\glsdetoklabel{#1}@currunit@\glsgetattribute{#1}{unitcount}.%
2943   \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}%
2944 }
```


previousunitcount

```
2945 \newcommand*\@glsxtr@previousunitcount[1]{%
2946   glo@glstetoklabel{#1}@prevunit@glstgetattribute{#1}{unitcount}.%
2947   \csuse{glsxtr@theunit@glstgetattribute{#1}{unitcount}}%
2948 }
```

t@currunitcount

```
2949 \newcommand*\@gls@increment@currunitcount[1]{%
2950   \glshasattribute{#1}{unitcount}%
2951   {%
2952     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
2953     \ifcsundef{\@glsxtr@csname}%
2954     {%
2955       \csgdef{\@glsxtr@csname}{1}%
2956       \listcsxadd
2957         {glo@glstetoklabel{#1}@unitlist}%
2958         {\glstgetattribute{#1}{unitcount}.%
2959         \csuse{glsxtr@theunit@glstgetattribute{#1}{unitcount}}%
2960         }%
2961     }%
2962   }%
2963   \csxdef{\@glsxtr@csname}%
2964     {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
2965   }%
2966 }%
2967 {}%
2968 }
```

t@currunitcount

```
2969 \newcommand*\@gls@local@increment@currunitcount[1]{%
2970   \glshasattribute{#1}{unitcount}%
2971   {%
2972     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
2973     \ifcsundef{\@glsxtr@csname}%
2974     {%
2975       \csdef{\@glsxtr@csname}{1}%
2976       \listcseadd
2977         {glo@glstetoklabel{#1}@unitlist}%
2978         {\glstgetattribute{#1}{unitcount}.%
2979         \csuse{glsxtr@theunit@glstgetattribute{#1}{unitcount}}%
2980         }%
2981     }%
2982   }%
2983   \csedef{\@glsxtr@csname}%
2984     {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
2985   }%
2986 }%
2987 {}%
2988 }
```

r@currunitcount

```
2989 \newcommand*{\@glsxtr@currunitcount}[2]{%
2990   \ifcsundef
2991   {glo@glstetoklabel{##1}@currunit@#2}%
2992   {0}%
2993   {\csuse{glo@glstetoklabel{##1}@currunit@#2}}%
2994 }%
```

r@prevunitcount

```
2995 \newcommand*{\@glsxtr@prevunitcount}[2]{%
2996   \ifcsundef
2997   {glo@glstetoklabel{##1}@prevunit@#2}%
2998   {0}%
2999   {\csuse{glo@glstetoklabel{##1}@prevunit@#2}}%
3000 }%
```

eentryunitcount

```
3001 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
3002   \appto\@newglossaryentry@defcounters{\@@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
3003   \renewcommand*{\gls@defdocnewglossaryentry}{%
3004     \renewcommand*\newglossaryentry[2]{%
3005       \PackageError{glossaries}{\string\newglossaryentry\space
3006         may only be used in the preamble when entry counting has
3007         been activated}{If you use \string\glsenableentryunitcount\space
3008         you must place all entry definitions in the preamble not in
3009         the document environment}%
3010     }%
3011   }%
  New commands to access new fields:
3012   \newcommand*{\glsentrycurrcount}[1]{%
3013     \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
3014     \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
3015   }%
3016   \newcommand*{\glsentryprevcount}[1]{%
3017     \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
3018     \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
3019   }%
  Access total count:
3020   \newcommand*{\glsentryprevtotalcount}[1]{%
3021     \ifcsundef{glo@glstetoklabel{##1}@prevunittotal}%
3022     {0}%
3023     {%
3024       \number\csuse{glo@glstetoklabel{##1}@prevunittotal}
3025     }%
3026   }%
```

Access max value:

```
3027 \newcommand*\glstentryprevmaxcount}[1]{%
3028 \ifcsundef{glo\glstetoklabel{##1}@prevunitmax}%
3029 {0}%
3030 {%
3031 \number\csuse{glo\glstetoklabel{##1}@prevunitmax}
3032 }%
3033 }%
```

Adjust post unset and reset:

```
3034 \let\@glstxtr@entryunitcount@org@unset\glstxtrpostunset
3035 \renewcommand*\glstxtrpostunset}[1]{%
3036 \@glstxtr@entryunitcount@org@unset{##1}%
3037 \@glst@increment@currunitcount{##1}%
3038 }%
3039 \let\@glstxtr@entryunitcount@org@localunset\glstxtrpostlocalunset
3040 \renewcommand*\glstxtrpostlocalunset}[1]{%
3041 \@glstxtr@entryunitcount@org@localunset{##1}%
3042 \@glst@local@increment@currunitcount{##1}%
3043 }%
3044 \let\@glstxtr@entryunitcount@org@reset\glstxtrpostreset
3045 \renewcommand*\glstxtrpostreset}[1]{%
3046 \glshasattribute{##1}{unitcount}%
3047 {%
3048 \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
3049 \ifcsundef{\@glstxtr@csname}%
3050 {}%
3051 {\csgdef{\@glstxtr@csname}{0}}%
3052 }%
3053 {}%
3054 }%
3055 \let\@glstxtr@entryunitcount@org@localreset\glstxtrpostlocalreset
3056 \renewcommand*\glstxtrpostlocalreset}[1]{%
3057 \@glstxtr@entryunitcount@org@localreset{##1}%
3058 \glshasattribute{##1}{unitcount}%
3059 {%
3060 \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
3061 \ifcsundef{\@glstxtr@csname}%
3062 {}%
3063 {\csdef{\@glstxtr@csname}{0}}%
3064 }%
3065 {}%
3066 }%
```

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

```
3067 \let\@cglst@\@cglst@
3068 \let\@cglstpl@\@cglstpl@
3069 \let\@cGlst@\@cGlst@
```

```

3070 \let\@cGlspl@\@cGlspl@
3071 \let\@cGLS@\@cGLS@
3072 \let\@cGLSpl@\@cGLSpl@

```

Write information to the aux file.

```

3073 \AtEndDocument{\@gls@write@entryunitcounts}%
3074 \renewcommand*{\@gls@entry@unitcount}[3]{%
3075   \csgdef{glo@glsdetoklabel{##1}@prevunit@##3}{##2}%
3076   \ifcsundef{glo@glsdetoklabel{##1}@prevunittotal}%
3077   {\csgdef{glo@glsdetoklabel{##1}@prevunittotal}{##2}}%
3078   {%
3079     \csxdef{glo@glsdetoklabel{##1}@prevunittotal}{
3080       \number\numexpr\csuse{glo@glsdetoklabel{##1}@prevunittotal}+##2}%
3081     }%
3082     \ifcsundef{glo@glsdetoklabel{##1}@prevunitmax}%
3083     {\csgdef{glo@glsdetoklabel{##1}@prevunitmax}{##2}}%
3084     {%
3085       \ifnum\csuse{glo@glsdetoklabel{##1}@prevunitmax}<##2
3086       \csgdef{glo@glsdetoklabel{##1}@prevunitmax}{##2}%
3087       \fi
3088     }%
3089   }%
3090 \let\glsenableentryunitcount\relax
3091 \renewcommand*{\glsenableentrycount}{%
3092   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
3093     can't be used with \string\glsenableentryunitcount}%
3094   {Use one or other but not both commands}%
3095 }%
3096 }
3097 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

3098 \newcommand*{\@gls@entry@unitcount}[3]{%

```

ryunitcounts@do

```

3099 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
3100   \immediate\write\@auxout
3101   {\string\@gls@entry@unitcount
3102     {\@glsentry}%
3103     {\@glsxtr@currunitcount{\@glsentry}{#1}%
3104     }%
3105     {#1}}%
3106 }

```

entryunitcounts

```

3107 \newcommand*{\@gls@write@entryunitcounts}{%
3108   \immediate\write\@auxout
3109   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}%
3110   \count@=0\relax

```

```

3111 \forallglsentries{\@glsentry}{%
3112   \glsattribute{\@glsentry}{unitcount}%
3113   {%
3114     \ifglsused{\@glsentry}%
3115     {%
3116       \forlistcsloop
3117         {\@gls@write@entryunitcounts@do}%
3118         {glo@glsdetoklabel{\@glsentry}@unitlist}%
3119     }%
3120   }%
3121   \advance\count@ by \@ne
3122 }%
3123 {}%
3124 }%
3125 \ifnum\count@=0
3126   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3127   \MessageBreak with \string\glsenableentryunitcount\space but the
3128   \MessageBreak attribute ‘unitcount’ hasn’t
3129   \MessageBreak been assigned to any of the defined
3130   \MessageBreak entries}%
3131 \fi
3132 }

```

`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.

```
3133 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
```

Enable entry counting:

```
3134 \glsenableentryunitcount
```

Redefine `\gls` etc:

```

3135 \renewcommand*{\gls}{\cgl}%
3136 \renewcommand*{\Gls}{\cGls}%
3137 \renewcommand*{\glspl}{\cglspl}%
3138 \renewcommand*{\Glspl}{\cGlspl}%
3139 \renewcommand*{\GLS}{\cGLS}%
3140 \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```
3141 \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%
```

In case this command is used again:

```

3142 \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
3143 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
3144   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
3145   can’t be used with \string\GlsXtrEnableEntryUnitCounting}%
3146   {Use one or other but not both commands}}%
3147 }

```

`countunsetattr`

```

3148 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
3149 \@for\@glsxtr@cat:=#1\do
3150 {%
3151   \ifdefempty{\@glsxtr@cat}{}%
3152   {%
3153     \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3154     \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
3155   }%
3156 }%
3157 }

```

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.

`\SetGenericNewAcronym`

```

3158 \renewcommand*{\SetGenericNewAcronym}{%
3159   \let\@Gls@entryname\@Gls@acrenryname
3160   \renewcommand{\newacronym}[4][ ]{%
3161     \ifdefempty{\@glsacronymlists}%
3162     {%
3163       \def\@glo@type{\acronymtype}%
3164       \setkeys{glossentry}{##1}%
3165       \DeclareAcronymList{\@glo@type}%
3166     }%
3167   }%
3168   \glskeylisttok{##1}%
3169   \glslabeltok{##2}%
3170   \glsshorttok{##3}%
3171   \glslongtok{##4}%
3172   \newacronymhook
3173   \protected@edef\@do@newglossaryentry{%
3174     \noexpand\newglossaryentry{\the\glslabeltok}%
3175     {%
3176       type=\acronymtype,%
3177       name={\expandonce{\acronymentry{##2}}},%
3178       sort={\acronymssort{\the\glsshorttok}{\the\glslongtok}},%
3179       text={\the\glsshorttok},%
3180       short={\the\glsshorttok},%
3181       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3182       long={\the\glslongtok},%
3183       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3184       category=acronym,

```

```

3185     \GenericAcronymFields,%
3186     \the\glskeylisttok
3187 }%
3188 }%
3189 \@do@newglossaryentry
3190 }%
3191 \renewcommand*{\acrfullfmt}[3]{%
3192   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
3193 \renewcommand*{\Acrfullfmt}[3]{%
3194   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
3195 \renewcommand*{\ACRfullfmt}[3]{%
3196   \glslink[##1]{##2}{%
3197     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}}%
3198 \renewcommand*{\acrfullplfmt}[3]{%
3199   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
3200 \renewcommand*{\Acrfullplfmt}[3]{%
3201   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
3202 \renewcommand*{\ACRfullplfmt}[3]{%
3203   \glslink[##1]{##2}{%
3204     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}}%
3205 \renewcommand*{\glstentryfull}[1]{\genacrfullformat{##1}{}}%
3206 \renewcommand*{\Glstentryfull}[1]{\Genacrfullformat{##1}{}}%
3207 \renewcommand*{\glstentryfullpl}[1]{\genplacrfullformat{##1}{}}%
3208 \renewcommand*{\Glstentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
3209 }

```

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:

```

3210 \let\@glstr@org@setacronymstyle\setacronymstyle
3211 \let\@glstr@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`.

```

3212 \newcommand*{\MakeAcronymsAbbreviations}{%
3213   \renewcommand*{\newacronym}[4][]{%
3214     \glstr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
3215   }%
3216   \renewcommand*{\firstacronymfont}[1]{\glstfirstabbrvfont{##1}}%
3217   \renewcommand*{\acronymfont}[1]{\glstabbrvfont{##1}}%
3218   \renewcommand*{\setacronymstyle}[1]{%
3219     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}
3220     unavailable.
3221     Use \string\setabbreviationstyle\space instead.
3222     The original acronym interface can be restored with
3223     \string\RestoreAcronyms}{}%
3224 }%
3225 \renewcommand*{\newacronymstyle}[1]{%

```

```

3226     \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
3227     available unless you restore the original acronym interface with
3228     \string\RestoreAcronyms}%
3229     \@glsxtr@org@newacronymstyle{##1}%
3230 }%
3231 }

```

Switch acronyms to abbreviations:

```
3232 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```

3233 \newcommand*\RestoreAcronyms}{%
3234   \SetGenericNewAcronym
3235   \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
3236   \renewcommand{\acronymfont}[1]{##1}%
3237   \let\setacronymstyle\@glsxtr@org@setacronymstyle
3238   \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.

```

3239 \renewcommand*\@gls@link@checkfirsthyper{%
3240   \ifglsused{\glslabel}%
3241   {\let\glsxtrifwasfirstuse\@secondoftwo}
3242   {\let\glsxtrifwasfirstuse\@firstoftwo}%
3243   \@glsxtr@org@checkfirsthyper
3244 }
3245 \glssetcategoryattribute{acronym}{regular}{false}%
3246 \setacronymstyle{long-short}%
3247 }

```

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

```

3248 \renewcommand*\glsacspace}[1]{%
3249   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
3250   \ifdim\dimen@<\glsacspacemax~\else\space\fi
3251 }

```

`\glsacspacemax` Value used in the above.

```
3252 \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`

```
3253 \newcommand*\@glsxtr@reg@glosslist}{}
```


Save the original definition of `\makeglossaries`:

```
3254 \let\@glsxtr@org@makeglossaries\makeglossaries
```

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

`\makeglossaries`

```
3255 \renewcommand*{\makeglossaries}[1] [] {%
3256   \ifblank{#1}%
3257   {\@glsxtr@org@makeglossaries}%
3258   {%
3259     \edef\@glsxtr@reg@glosslist{#1}%
3260     \ifundef{\glswrite}{\newwrite\glswrite}{}%
3261     \protected@write\@auxout{}{\string\providecommand
3262       \string\@glsorder[1]{}%
3263     \protected@write\@auxout{}{\string\providecommand
3264       \string\@istfilename[1]{}%
3265     \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
3266     \protected@write\@auxout{}{\string\@glsorder{\glsorder}}%
3267     \protected@write\@auxout{}{\string\@glsxtr@makeglossaries{#1}}%
3268     \write\@auxout{\string\providecommand\string\@gls@reference[3]{}%}
```

Iterate through each supplied glossary type and activate it.

```
3269   \@for\@glo@type:=#1\do{%
3270     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
3271   }%
```

New glossaries must be created before `\makeglossaries`:

```
3272 \renewcommand*\newglossary[4] [] {%
3273 \PackageError{glossaries}{New glossaries
3274 must be created before \string\makeglossaries}{You need
3275 to move \string\makeglossaries\space after all your
3276 \string\newglossary\space commands}}%
```

Any subsequent instances of this command should have no effect

```
3277 \let\@makeglossary\relax
3278 \let\makeglossary\relax
3279 \renewcommand\makeglossaries[1] [] {}%
```

Disable all commands that have no effect after `\makeglossaries`

```
3280 \@disable@onlypremakeg
```

Allow see key:

```
3281 \let\gls@checkseeallowed\relax
```

Adjust `\@do@seeglossary`

```
3282 \renewcommand*{\@do@seeglossary}[2] {%
3283   \edef\@gls@label{\glsdetoklabel{##1}}%
3284   \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3285   \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%}
```

```

3286   {\@glsxtr@org@doseeglossary{##1}{##2}}%
3287   {%
3288     \protected@write\@auxout{}{%
3289       \string\@gls@reference
3290       {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
3291     }%
3292   }%
3293 }%

Adjust \@do@wrglossary
3294 \let\@glsxtr@do@wrglossary\@do@wrglossary
3295 \def\@do@wrglossary{%
3296   \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3297   \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3298   {\@glsxtr@do@wrglossary}%
3299   {\gls@noidxglossary}%
3300 }%

Suppress warning about no \makeglossaries
3301 \let\warn@nomakeglossaries\relax
3302 \def\warn@noprntglossary{%
3303   \GlossariesWarningNoLine{No \string\printglossary\space
3304     or \string\printglossaries\space
3305     found.^^J(Remove \string\makeglossaries\space if you don't want
3306     any glossaries.)^^JThis document will not have a glossary}%
3307 }%

Only warn for glossaries not listed.
3308 \renewcommand{\@gls@noref@warn}[1]{%
3309   \edef\@gls@type{##1}%
3310   \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3311   {%
3312     \GlossariesExtraWarning{Can't use
3313       \string\printnoidxglossary[type={\@gls@type}]
3314       when '\@gls@type' is listed in the optional argument of
3315       \string\makeglossaries}%
3316   }%
3317   {%
3318     \GlossariesWarning{Empty glossary for
3319       \string\printnoidxglossary[type={##1}].
3320       Rerun may be required (or you may have forgotten to use
3321       commands like \string\gls)}%
3322   }%
3323 }%

Adjust display number list to check for type:
3324 \renewcommand*\@glsdisplaynumberlist}[1]{%
3325   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3326   {\@glsxtr@idx@displaynumberlist{##1}}%
3327   {\@glsxtr@noidx@displaynumberlist{##1}}%
3328 }%

```

Adjust entry list:

```
3329 \renewcommand*\glsentrynumberlist}[1]{%
3330   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3331   {\@glsxtr@idx@entrynumberlist{##1}}%
3332   {\@glsxtr@noidx@entrynumberlist{##1}}%
3333 }%
```

Adjust number list loop

```
3334 \renewcommand*\glsnumberlistloop}[2]{%
3335   \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3336   {%
3337     \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3338     not available for glossary ‘##1’}{}%
3339   }%
3340   {\@glsxtr@noidx@numberlistloop{##1}{##2}}%
3341 }%
```

Only sanitize sort for normal indexing glossaries.

```
3342 \renewcommand*\glsprestandardsort}[3]{%
3343   \expandafter\DTLifinlist\expandafter{##2}{\@glsxtr@reg@glosslist}%
3344   {%
3345     \glsdosanitizesort
3346   }%
3347   {%
3348     \ifglssanitizesort
3349     \@gls@noidx@sanitizesort
3350   \else
3351     \@gls@noidx@nosanitizesort
3352   \fi
3353 }%
3354 }%
```

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

```
3355 \renewcommand*\new@glossaryentry[2]{%
3356   \PackageError{glossaries-extra}{Glossary entries must be defined
3357   in the preamble\MessageBreak when you use the optional argument
3358   of \string\makeglossaries}{Either move your definitions to the
3359   preamble or don't use the optional argument of
3360   \string\makeglossaries}%
3361 }%
```

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).

```
3362 \let\@glo@assign@sortkey\@glsxtr@mixed@assign@sortkey
3363 \renewcommand*\@printgloss@setsort}{%
```

Need to extract just the type value.

```
3364   \expandafter\@glsxtr@gettype\expandafter,\@glsxtr@printglossopts,%
3365   type=\glsdefaulttype,\@end@glsxtr@gettype
3366   \def\@glo@sorttype{\@glo@default@sorttype}%
```

3367 }%

Check automake setting:

```
3368 \ifglsautomake
3369   \renewcommand*{\@gls@doautomake}{%
3370     \@for\@gls@type:=\@glsxtr@reg@glosslist\do{%
3371       \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3372     }%
3373   }%
3374 \fi
```

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

```
3375 \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
3376 }%
3377 }
```

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

`rgprintglossary` 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 `gls` file for some settings, so the glossary might not have been defined.) (This command is also used for on-the-fly setting.)

```
3378 \newcommand{\@glsxtr@orgprintglossary}[2]{%
3379   \def\@glo@type{\glsdefaulttype}%
```

Add check here.

```
3380 \def\glossarytitle{%
3381   \ifcsdef{@glo@type@\@glo@type @title}%
3382     {\csuse{@glo@type@\@glo@type @title}}%
3383     {\glossaryname}}%
3384 \def\glossarytoctitle{\glossarytitle}%
3385 \let\org@glossarytitle\glossarytitle
3386 \def\@glossarystyle{%
3387   \ifx\@glossary@default@style\relax
3388     \GlossariesWarning{No default glossary style provided \MessageBreak
3389       for the glossary '@@glo@type'. \MessageBreak
3390       Using deprecated fallback. \MessageBreak
3391       To fix this set the style with \MessageBreak
3392       \string\setglossarystyle\space or use the \MessageBreak
3393       style key=value option}%
3394   \fi
3395 }%
3396 \def\gls@dotoc@title{\gls@set@toc@title{\@glo@type}}%
3397 \let\@org@glossaryentrynumbers\glossaryentrynumbers
3398 \bgroup
3399   \@printgloss@setsort
3400   \setkeys{printgloss}{#1}%
3401   \ifx\glossarytitle\org@glossarytitle
3402   \else
```

```

3403     \cslet{@glo@type@\glo@type @title}{\glossarytitle}%
3404     \fi
3405     \let\currentglossary\@glo@type
3406     \let\org@glossaryentrynumbers\glossaryentrynumbers
3407     \let\glsnonextpages\@glsnonextpages
3408     \let\glsnextpages\@glsnextpages
3409     \let\nopostdesc\@nopostdesc
3410     \gls@dotoc@title
3411     \@glossarystyle
3412     \let\gls@org@glossaryentryfield\glossentry
3413     \let\gls@org@glossarysubentryfield\subglossentry
3414     \renewcommand{\glossentry}[1]{%
3415         \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3416         \gls@org@glossaryentryfield{##1}%
3417     }%
3418     \renewcommand{\subglossentry}[2]{%
3419         \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3420         \gls@org@glossarysubentryfield{##1}{##2}%
3421     }%
3422     \@gls@preglossaryhook
3423     #2%
3424     \egroup
3425     \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
3426     \global\let\warn@noprintglossary\relax
3427 }

```

`\@printglossary` Redefine.

```

3428 \renewcommand{\@printglossary}[2]{%
3429     \def\@glsxtr@printglossopts{##1}%
3430     \@glsxtr@orgprintglossary{##1}{##2}%
3431 }

```

Add a key that switches off the entry targets:

```

3432 \define@choicekey{printgloss}{target}[\val\nr]{true,false}[true]{%
3433     \ifcase\nr
3434         \let\@gls@target\glsdohypertarget
3435     \else
3436         \let\@gls@target\@secondoftwo
3437     \fi
3438 }

```

`hypernameprefix`

```

3439 \newcommand{\@glsxtrhypernameprefix}{}

```

New to v1.20:

```

3440 \define@key{printgloss}{targetnameprefix}{%
3441     \renewcommand{\@glsxtrhypernameprefix}{##1}%
3442 }

```

glsdohypertarget Redefine to insert \@glsxtrhypernameprefix before the target name.

```
3443 \let\@glsxtr@org@glsdohypertarget@glsdohypertarget
3444 \renewcommand{\glsdohypertarget}[2]{%
3445   \@glsxtr@org@glsdohypertarget{\@glsxtrhypernameprefix#1}{#2}%
3446 }
```

@makeglossaries For the benefit of makeglossaries

```
3447 \newcommand*\@glsxtr@makeglossaries}[1]{}
```

@glsxtr@gettype Get just the type.

```
3448 \def\@glsxtr@gettype#1,type=#2,#3\@end@glsxtr@gettype{%
3449   \def\@glo@type{#2}%
3450 }
```

@assign@sortkey Assign the sort key.

```
3451 \newcommand\@glsxtr@mixed@assign@sortkey[1]{%
3452   \edef\@glo@type{\@glo@type}%
3453   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glsxtr@reg@glosslist}%
3454   {%
3455     \@glo@no@assign@sortkey{#1}%
3456   }%
3457   {%
3458     \@glo@assign@sortkey{#1}%
3459   }%
3460 }%
```

Display number list for the regular version:

splaynumberlist

```
3461 \let\@glsxtr@idx@displaynumberlist@glsdisplaynumberlist
```

Display number list for the “noidx” version:

splaynumberlist

```
3462 \newcommand*\@glsxtr@noidx@displaynumberlist}[1]{%
3463   \letcs{\@gls@loclist}{glo@glsdetoklabel{#1}@loclist}%
3464   \ifdef\@gls@loclist
3465     {%
3466       \def\@gls@noidxloclist@sep{%
3467         \def\@gls@noidxloclist@sep{%
3468           \def\@gls@noidxloclist@sep{%
3469             \glsnumlistsep
3470           }%
3471           \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
3472         }%
3473       }%
3474       \def\@gls@noidxloclist@finalsep{}%
3475       \def\@gls@noidxloclist@prev{}%
3476       \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%

```

```

3477 \@gls@noidxloclist@finalsep
3478 \@gls@noidxloclist@prev
3479 }%
3480 {%

3481 \glsxtrundeftag
3482 \glsdoifexists{#1}%
3483 {%
3484 \GlossariesWarning{Missing location list for ‘#1’. Either
3485     a rerun is required or you haven’t referenced the entry.}%
3486 }%
3487 }%
3488 }%
3489

```

And for the number list loop:

@numberlistloop

```

3490 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
3491 \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3492 \let\@gls@org@glsnoidxdisplayloc@glsnoidxdisplayloc
3493 \let\@gls@org@glsseeformat@glsseeformat
3494 \let\glsnoidxdisplayloc#2\relax
3495 \let\glsseeformat#3\relax
3496 \ifdef\@gls@loclist
3497 {%
3498 \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3499 }%
3500 {%

3501 \glsxtrundeftag
3502 \glsdoifexists{#1}%
3503 {%
3504 \GlossariesWarning{Missing location list for ‘##1’. Either
3505     a rerun is required or you haven’t referenced the entry.}%
3506 }%
3507 }%
3508 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3509 \let\glsseeformat\@gls@org@glsseeformat
3510 }%

```

Same for entry number list.

entrynumberlist

```

3511 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
3512 \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3513 \ifdef\@gls@loclist
3514 {%
3515 \glsnoidxloclist{\@gls@loclist}%
3516 }%
3517 {%

```

```

3518 \glxtrundeftag
3519 \glsdoifexists{#1}%
3520 {%
3521 \GlossariesWarning{Missing location list for ‘#1’. Either
3522 a rerun is required or you haven’t referenced the entry.}%
3523 }%
3524 }%
3525 }%

```

entrynumberlist

```

3526 \newcommand*{\@glxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

3527 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
3528 \protected@edef\@glxtr@titlelabel{#1}%
3529 \ifdefvoid\@glxtr@titlelabel
3530 {}%
3531 {%
3532 \protected@edef\@glxtr@titlelabel{\csuse{glxtr@grouptitle@#1}}%
3533 }%
3534 \ifdefvoid{\@glxtr@titlelabel}%
3535 {%
3536 \DTLifint{#1}%
3537 {%
3538 \ifnum#1<256\relax
3539 \edef#2{\char#1\relax}%
3540 \else
3541 \edef#2{#1}%
3542 \fi
3543 }%
3544 {%
3545 \ifcsundef{#1groupname}%
3546 {\def#2{#1}}%
3547 {\letcs#2{#1groupname}}%
3548 }%
3549 }%
3550 {%
3551 \let#2\@glxtr@titlelabel
3552 }%
3553 }

```

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

```

3554 \let\glxtr@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.

```

3555 \newrobustcmd{\glxtrgetgrouptitle}[2]{%
3556 \protected@edef\@glxtr@titlelabel{glxtr@grouptitle@#1}%
3557 \@onelevel@sanitize\@glxtr@titlelabel

```



```

3558 \ifcsdef{\@glsxtr@titlelabel}
3559 {\letcs{#2}{\@glsxtr@titlelabel}}%
3560 {\glsxtr@org@getgrouptitle{#1}{#2}}%
3561 }
3562 \let\@gls@getgrouptitle\glsxtrgetgrouptitle

```

trsetgrouptitle Sets the title for the given group label.

```

3563 \newcommand{\glsxtrsetgrouptitle}[2]{%
3564 \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3565 \@onelevel@sanitize\@glsxtr@titlelabel
3566 \csxdef{\@glsxtr@titlelabel}{#2}%
3567 }

```

\glsnavigation Redefine to use new user-level command.

```

3568 \renewcommand*{\glsnavigation}{%
3569 \def\@gls@between{}%
3570 \ifcsundef{\@gls@hypergrouplist@\@glo@type}%
3571 {%
3572 \def\@gls@list{}%
3573 }%
3574 {%
3575 \expandafter\let\expandafter\@gls@list
3576 \csname \@gls@hypergrouplist@\@glo@type\endcsname
3577 }%
3578 \@for\@gls@tmp:=\@gls@list\do{%
3579 \@gls@between
3580 \glsxtrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
3581 \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
3582 \let\@gls@between\glshypernavsep
3583 }%
3584 }

```

@noidx@glossary

```

3585 \renewcommand*{\@print@noidx@glossary}{%
3586 \ifcsdef{\@glsref@\@glo@type}%
3587 {%
3588 \ifcsdef{\@glo@sortmacro@\@glo@sorttype}%
3589 {%
3590 \csuse{\@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
3591 }%
3592 {%
3593 \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
3594 }%
3595 \glossarysection[\glossarytoctitle]{\glossarytitle}%
3596 \glossary preamble

```

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

```

3597 \def\@gls@currentlettergroup{}%

```

```

3598   \begin{theglossary}%
3599   \glossaryheader
3600   \glsresetentrylist
3601   \forlistcsloop{\@gls@noidx@do}{\@glsref@\@glo@type}%
3602   \end{theglossary}%
3603   \glossarypostamble
3604 }%
3605 {%

```

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

```

3606   \glsxtrifemptyglossary{\@glo@type}%
3607   }%
3608   {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
3609   \@gls@noref@warn{\@glo@type}%
3610 }%
3611 }

```

`noidxdisplayloc` Patch to check for range formations.

```

3612 \renewcommand*\glsnoidxdisplayloc[4]{%
3613   \setentrycounter[#1]{#2}%
3614   \@glsxtr@display@loc#3\empty\end@glsxtr@display@loc{#4}%
3615 }

```

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

```

3616 \def\@glsxtr@display@loc#1#2\end@glsxtr@display@loc#3{%
3617   \ifx#1(\relax
3618     \glsxtrdisplaystartloc{#2}{#3}%
3619   \else
3620     \ifx#1)\relax
3621       \glsxtrdisplayendloc{#2}{#3}%
3622     \else
3623       \glsxtrdisplaysingleloc{#1#2}{#3}%
3624     \fi
3625   \fi
3626 }

```

`isplayingleloc` Single location.

```

3627 \newcommand*\glsxtrdisplaysingleloc[2]{%
3628   \csuse{#1}{#2}%
3629 }

```

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 `\glsxtrlocrangefmt`.

`displaystartloc` Start of a location range.

```

3630 \newcommand*\glsxtrdisplaystartloc[2]{%
3631   \edef\glsxtrlocrangefmt{#1}%
3632   \ifx\glsxtrlocrangefmt\empty

```

```

3633 \def\glxtrlocrangefmt{glsnumberformat}%
3634 \fi
3635 \expandafter\glxtrdisplaysingleloc
3636 \expandafter{\glxtrlocrangefmt}{#2}%
3637 }

```

trdisplayendloc End of a location range.

```

3638 \newcommand*\glxtrdisplayendloc}[2]{%
3639 \edef\@glxtr@tmp{#1}%
3640 \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{glsnumberformat}}{}%
3641 \ifx\glxtrlocrangefmt\@glxtr@tmp
3642 \else
3643 \GlossariesExtraWarning{Mismatched end location range
3644 (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
3645 \fi
3646 \expandafter\glxtrdisplayendloohook\expandafter{\@glxtr@tmp}{#2}%
3647 \expandafter\glxtrdisplaysingleloc
3648 \expandafter{\glxtrlocrangefmt}{#2}%
3649 \def\glxtrlocrangefmt{}%
3650 }

```

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

```

3651 \newcommand*\glxtrdisplayendloohook}[2]{

```

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

```

3652 \newcommand*\glxtrlocrangefmt{}

```

ls@removespaces Redefine to allow adjustments to location hyperlink.

```

3653 \def\@gls@removespaces#1 #2\@nil{%
3654 \toks@=\expandafter{\the\toks@#1}%
3655 \ifx\@#2\@%
3656 \edef\x{\the\toks@}%
3657 \ifx\x\empty
3658 \else
3659 \glxtrlocationhyperlink{\glsentrycounter}{\@glo@counterprefix}{\the\toks@}%
3660 \fi
3661 \else
3662 \@gls@ReturnAfterFi{%
3663 \@gls@removespaces#2\@nil
3664 }%
3665 \fi
3666 }

```

locationhyperlink

```

3667 \newcommand*\glxtrlocationhyperlink}[3]{%
3668 \ifdefvoid\glxtrsupplocationurl
3669 {%
3670 \glxtrhyperlink{#1#2#3}{#3}%
3671 }%

```

```

3672 {%
3673   \hyperref{\glxtrsupplocationurl}{\#1#2#3}{#3}%
3674 }%
3675 }

```

supphypernumber

```

3676 \newcommand*\glxtrsupphypernumber[1]{%
3677   {%
3678     \glshasattribute{\glscurrententrylabel}{externallocation}%
3679     {%
3680       \def\glxtrsupplocationurl{%
3681         \glsggetattribute{\glscurrententrylabel}{externallocation}}%
3682     }%
3683     {%
3684       \def\glxtrsupplocationurl{}%
3685     }%
3686     \glshypernumber{#1}%
3687   }%
3688 }

```

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

@print@glossary

```

3689 \renewcommand*\@print@glossary{%
3690   \makeatletter
3691   \@input@{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
3692   \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
3693   {}%
3694   {\glxtrNoGlossaryWarning{\@glo@type}}%
3695   \ifglxindy
3696     \ifcsundef{@xdy@\@glo@type @language}%
3697     {%
3698       \edef\@do@auxoutstuff{%
3699         \noexpand\AtEndDocument{%
3700           \noexpand\immediate\noexpand\write\@auxout{%
3701             \string\providecommand\string\@xdylanguage[2]{}}%
3702           \noexpand\immediate\noexpand\write\@auxout{%
3703             \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
3704         }%
3705       }%
3706     }%
3707     {%
3708       \edef\@do@auxoutstuff{%
3709         \noexpand\AtEndDocument{%
3710           \noexpand\immediate\noexpand\write\@auxout{%
3711             \string\providecommand\string\@xdylanguage[2]{}}%
3712           \noexpand\immediate\noexpand\write\@auxout{%
3713             \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type

```

```

3714         @language\endcsname}}%
3715     }%
3716 }%
3717 }%
3718 \do@auxoutstuff
3719 \edef\do@auxoutstuff{%
3720     \noexpand\AtEndDocument{%
3721         \noexpand\immediate\noexpand\write\@auxout{%
3722             \string\providecommand\string\@gls@codepage[2]{}}%
3723         \noexpand\immediate\noexpand\write\@auxout{%
3724             \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
3725     }%
3726 }%
3727 \do@auxoutstuff
3728 \fi
3729 \renewcommand*{\@warn@nomakeglossaries}{%
3730     \GlossariesWarningNoLine{\string\makeglossaries\space
3731     hasn't been used,^^Jthe glossaries will not be updated}%
3732 }%
3733 }

```

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

`oGlsWarningHead` Header message.

```

3734 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
3735     This document is incomplete. The external file associated with
3736     the glossary '#1' (which should be called \texttt{#2})
3737     hasn't been created.%
3738 }

```

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

```

3739 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
3740     This has probably happened because there are no entries defined
3741     in this glossary.%
3742 }

```

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

```

3743 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
3744     If you don't want this glossary,
3745     add \texttt{nomain} to your package option list when you load
3746     \texttt{glossaries-extra.sty}. For example:%
3747 }

```

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

```

3748 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
3749     Did you forget to use \texttt{type=#1} when you defined your
3750     entries? If you tried to load entries into this glossary with
3751     \texttt{\string\loadglsentries} did you remember to use
3752     \texttt{[#1]} as the optional argument? If you did, check that

```

```

3753 the definitions in the file you loaded all had the type set
3754 to \texttt{\string\glsdefaulttype}.%
3755 }

```

WarningCheckFile Advisory message to check the file contents.

```

3756 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
3757   Check the contents of the file \texttt{#1}. If
3758   it's empty, that means you haven't indexed any of your entries in this
3759   glossary (using commands like \texttt{\string\gls} or
3760   \texttt{\string\glsadd}) so this list can't be generated.
3761   If the file isn't empty, the document build process hasn't been
3762   completed.%
3763 }

```

WarningAutoMake Message when automake option has been used.

```

3764 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
3765   You may need to rerun \LaTeX. If you already have, it may be that
3766   \TeX's shell escape doesn't allow you to run
3767   \ifglxindy xindy\else makeindex\fi. Check the
3768   transcript file \texttt{\jobname.log}. If the shell escape is
3769   disabled, try one of the following:
3770
3771   \begin{itemize}
3772     \item Run the external (Lua) application:
3773
3774       \texttt{makeglossaries-lite.lua \string\jobname\string}
3775
3776     \item Run the external (Perl) application:
3777
3778       \texttt{makeglossaries \string\jobname\string}
3779   \end{itemize}
3780
3781   Then rerun \LaTeX\ on this document.
3782   \GlossariesExtraWarning{Rerun required to build the
3783   glossary '#1' or check TeX's shell escape allows
3784   you to run \ifglxindy xindy\else makeindex\fi}%
3785 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

3786 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
3787   You need to either replace \texttt{\string\makenoidxglossaries}
3788   with \texttt{\string\makeglossaries} or replace
3789   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
3790   \texttt{\string\printnoidxglossary}
3791   (or \texttt{\string\printnoidxglossaries}) and then rebuild
3792   this document.%
3793 }

```

arningBuildInfo Build advice.

```
3794 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
3795   Try one of the following:
3796   \begin{itemize}
3797     \item Add \texttt{automake} to your package option list when you load
3798       \texttt{glossaries-extra.sty}. For example:
3799
3800       \texttt{\string\usepackage[automake]%
3801         \glsopenbrace glossaries-extra\glsclosebrace}
3802
3803     \item Run the external (Lua) application:
3804
3805     \texttt{makeglossaries-lite.lua \string"\jobname\string"}
3806
3807     \item Run the external (Perl) application:
3808
3809     \texttt{makeglossaries \string"\jobname\string"}
3810   \end{itemize}
3811
3812   Then rerun \LaTeX\ on this document.%
3813 }
```

oGlsWarningTail Final paragraph.

```
3814 \newcommand{\GlsXtrNoGlsWarningTail}{%
3815   This message will be removed once the problem has been fixed.%
3816 }
```

GlsWarningNoOut No out file created. Build advice.

```
3817 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
3818   The file \texttt{#1} doesn't exist. This most likely means you haven't used
3819   \texttt{\string\makeglossaries} or you have used
3820   \texttt{\string\nofiles}. If this is just a draft version of the
3821   document, you can suppress this message using the
3822   \texttt{nomissingglstext} package option.%
3823 }
```

glossarywarning

```
3824 \newcommand*{@glsxtr@defaultnoglossarywarning}[1]{%
3825   \glossarysection[\glossarytoctitle]{\glossarytitle}
3826   \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glo@type @in\endcsname}
3827   \par
3828   \glsxtrifemptyglossary{#1}%
3829   {%
3830     \GlsXtrNoGlsWarningEmptyStart\space
3831     \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
3832     \medskip
3833     \noindent\texttt{\string\usepackage[nomain\ifglsacronym ,acronym\fi]%
3834       \glsopenbrace glossaries-extra\glsclosebrace}
3835     \medskip
```

```

3836 }%
3837 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
3838 }%
3839 {%
3840 \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
3841 {%
3842 \GlsXtrNoGlsWarningCheckFile
3843 {\jobname.\csname @glotype@\@glo@type @out\endcsname}
3844
3845 \ifglsautomake
3846
3847 \GlsXtrNoGlsWarningAutoMake{#1}
3848
3849 \else
3850
3851 \ifthenelse{\equal{#1}{main}}%
3852 {%
3853 \GlsXtrNoGlsWarningEmptyMain\par
3854 \medskip
3855 \noindent\texttt{\string\usepackage[nomain]%
3856 \glsopenbrace glossaries-extra\glsclosebrace}
3857 \medskip
3858 }%
3859 }%
3860
3861 \ifdefequal\makeglossaries\@no@makeglossaries
3862 {%
3863 \GlsXtrNoGlsWarningMisMatch
3864 }%
3865 }%
3866 \GlsXtrNoGlsWarningBuildInfo
3867 }%
3868 \fi
3869 }%
3870 }%
3871 \GlsXtrNoGlsWarningNoOut
3872 {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
3873 }%
3874 }%
3875 \par
3876 \GlsXtrNoGlsWarningTail
3877 }

```

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

`xtrresourcefile` 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.

```

3878 \newcommand*\glxtrresourcefile[2] [] {%
3879 \glxtr@writefields
3880 \protected@write\@auxout{}{\string\glxtr@resource{#1}{#2}}%

```



```

3881 \let\@glxtr@org@see@noindex\@glx@see@noindex
3882 \let\@glx@see@noindex\relax
3883 \IfFileExists{#2.glstex}%
3884 {%

```

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

```

3885 \edef\@bibglx@restreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
3886 \makeatletter
3887 \@input{#2.glstex}%
3888 \@bibglx@restreat
3889 }%
3890 {%
3891 \GlossariesExtraWarning{No file '#2.glstex'}%
3892 }%
3893 \let\@glx@see@noindex\@glxtr@org@see@noindex
3894 }
3895 \@onlypreamble\glxtrresourcefile

```

trresourcecount

```

3896 \newcount\glxtrresourcecount

```

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

```

3897 \newcommand*\GlsXtrLoadResources[1][{}]{%
3898 \ifnum\glxtrresourcecount=0\relax
3899 \glxtrresourcefile[#1]{\jobname}%
3900 \else
3901 \glxtrresourcefile[#1]{\jobname-\the\glxtrresourcecount}%
3902 \fi
3903 \advance\glxtrresourcecount by 1\relax
3904 }

```

glxtr@resource

```

3905 \newcommand*\glxtr@resource[2]{}

```

\glxtr@fields

```

3906 \newcommand*\glxtr@fields[1]{}

```

xtr@texencoding

```

3907 \newcommand*\glxtr@texencoding[1]{}

```

\glxtr@langtag

```

3908 \newcommand*\glxtr@langtag[1]{}

```

@pluralsuffixes

```

3909 \newcommand*\glxtr@pluralsuffixes[4]{}

```

tr@shortcutsval

```

3910 \newcommand*\glxtr@shortcutsval[1]{}

```

sxtr@linkprefix

```
3911 \newcommand*\glsxtr@linkprefix}[1]{}
```

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

```
3912 \newcommand*\glsxtr@writefields{%
```

```
3913 \protected@write\@auxout{%
```

```
3914 {\string\providecommand*\string\glsxtr@fields}[1]{}}%
```

```
3915 \protected@write\@auxout{%
```

```
3916 {\string\providecommand*\string\glsxtr@resource}[2]{}}%
```

```
3917 \protected@write\@auxout{%
```

```
3918 {\string\providecommand*\string\glsxtr@pluralsuffixes}[4]{}}%
```

```
3919 \protected@write\@auxout{%
```

```
3920 {\string\providecommand*\string\glsxtr@shortcutsval}[1]{}}%
```

```
3921 \protected@write\@auxout{%
```

```
3922 {\string\providecommand*\string\glsxtr@linkprefix}[1]{}}%
```

```
3923 \protected@write\@auxout{\string\glsxtr@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 `\glsxtrresourcefile`.

```
3924 \ifdef\CurrentTrackedLanguageTag
```

```
3925 {%
```

```
3926 \protected@write\@auxout{%
```

```
3927 \string\glsxtr@langtag{\CurrentTrackedLanguageTag}}%
```

```
3928 }%
```

```
3929 {}%
```

```
3930 \protected@write\@auxout{\string\glsxtr@pluralsuffixes
```

```
3931 {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}}%
```

```
3932 {\glxtrabbrvpluralsuffix}}%
```

```
3933 \ifdef\inputencodingname
```

```
3934 {%
```

```
3935 \protected@write\@auxout{\string\glsxtr@texencoding{\inputencodingname}}%
```

```
3936 }%
```

```
3937 {%
```

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.)

```
3938 \@ifpackageloaded{fontspec}%
```

```
3939 {\protected@write\@auxout{\string\glsxtr@texencoding{utf8}}}%
```

```
3940 {}%
```

```
3941 }%
```

```
3942 \protected@write\@auxout{\string\glsxtr@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.

```
3943 \AtBeginDocument
```

```
3944 {\protected@write\@auxout{\string\glsxtr@linkprefix{\glolinkprefix}}}%
```

```
3945 \let\glsxtr@writefields\relax
```

If the automake option is on, try running bib2gls if the aux file exists. The double-quotes around \jobname have been removed (v1.19) since \jobname will include double-quotes if the file name has spaces.

```
3946 \ifglsautomake
3947 \IfFileExists{\jobname.aux}%
3948 {\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.

```
3949 \ifx\@gls@doautomake\@gls@doautomake@err
3950 \let\@gls@doautomake\relax
3951 \fi
3952 \fi
3953 }
```

do@automake@err

```
3954 \newcommand*{\@gls@doautomake@err}{%
3955 \PackageError{glossaries}{You must use
3956 \string\makeglossaries\space with automake=true}
3957 {%
3958 Either remove the automake=true setting or
3959 add \string\makeglossaries\space to your document preamble.%
3960 }%
3961 }
```

Allow locations specific to a particular counter to be recorded.

\glsxtr@record

```
3962 \newcommand*{\glsxtr@record}[5] {}
```

r@counterrecord Aux file command.

```
3963 \newcommand*{\glsxtr@counterrecord}[3]{%
3964 \glsxtrfieldlistgadd{#1}{record.#2}{#3}%
3965 }
```

unterrecordhook Hook used by \@glsxtr@dorecord.

```
3966 \newcommand*{\@glsxtr@counterrecordhook} {}
```

trRecordCounter Activate recording for a particular counter (identified in the argument).

```
3967 \newcommand*{\GlsXtrRecordCounter}[1]{%
3968 \@glsxtr@recordcounter{#1}%
3969 }
3970 \@onlypreamble\GlsXtrRecordCounter
```

docounterrecord

```
3971 \newcommand*{\@glsxtr@docounterrecord}[1]{%
3972 \protected@write\@auxout{}{\string\glsxtr@counterrecord
3973 {\@gls@label}{#1}{\csuse{the#1}}}%
3974 }
```

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

```
3975 \newcommand*\printunsrtglossary}{%
3976   \ifstar\s@printunsrtglossary\@printunsrtglossary
3977 }
```

ntunsrtglossary Unstarred version.

```
3978 \newcommand*\@printunsrtglossary}[1][ ]{%
3979   \@printglossary{type=\glsdefaulttype,#1}\@print@unsrt@glossary}%
3980 }
```

ntunsrtglossary Starred version.

```
3981 \newcommand*\s@printunsrtglossary}[2][ ]{%
3982   \begingroup
3983     #2%
3984     \@printglossary{type=\glsdefaulttype,#1}\@print@unsrt@glossary}%
3985   \endgroup
3986 }
```

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

```
3987 \newcommand*\printunsrtglossaries}{%
3988   \forallglossaries{\@glo@type}\printunsrtglossary[type=\@glo@type]}%
3989 }
```

@unsrt@glossary

```
3990 \newcommand*\@print@unsrt@glossary}{%
3991   \glossarysection[\glossarytoctitle]{\glossarytitle}%
3992   \glossarypreamble
3993   check for empty list
3994   \glsxtrifemptyglossary{\@glo@type}%
3995   {%
3996     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
3997   }%
3998   \key@ifundefined{glossentry}{group}%
3999   {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
4000   {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
4001   \def\@gls@currentlettergroup{}}
```

A loop within the tabular-like styles can cause problems, so move the loop outside.

```
4002 \def\@glsxtr@doglossary{%
4003   \begin{theglossary}%
4004     \glossaryheader
4005     \glsresetentrylist
4006   }%
4007   \expandafter\@for\expandafter\glscurrententrylabel\expandafter
```

```

4008      :\expandafter=\csname glolist@\@glo@type\endcsname\do{%
4009      \ifdefempty{\glscurrententrylabel}
4010      }{%
4011      {\eappto\@glxtr@doglossary{%
4012      \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}}%
4013      }%
4014      \appto\@glxtr@doglossary{\end{theglossary}}%
4015      \@glxtr@doglossary
4016      }%
4017      \glossarypostamble
4018      }

```

lossary@handler

```

4019 \newcommand{\@printunsrt@glossary@handler}[1]{%
4020 \xdef\glscurrententrylabel{#1}%
4021 \printunsrtglossaryhandler\glscurrententrylabel
4022 }

```

glossaryhandler

```

4023 \newcommand{\printunsrtglossaryhandler}[1]{%
4024 \glxtrunsrtdo{#1}%
4025 }

```

srtglossaryunit

```

4026 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
4027 \s@printunsrtglossary[type=\glstypedefaulttype,#1]{%
4028 \printunsrtglossaryunitsetup{#2}%
4029 }%
4030 }

```

ossaryunitsetup

```

4031 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
4032 \renewcommand{\printunsrtglossaryhandler}[1]{%
4033 \glxtrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
4034 {\glxtrunsrtdo{##1}}%
4035 }%
4036 }%

```

Only the target names should have the prefixes adjusted as \gls etc need the original \glolinkprefix. The \@gobble part discards \glolinkprefix.

```

4037 \ifcsundef{theH#1}%
4038 {%
4039 \renewcommand*{\@glxtrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
4040 }%
4041 {%
4042 \renewcommand*{\@glxtrhypernameprefix}{record.#1.\csuse{theH#1}.\@gobble}%
4043 }%
4044 \renewcommand*{\glossarysection}[2][ ]{%
4045 \appto\glossarypostamble{\glspare\medskip\glspare}%
4046 }

```

srtglossaryunit

```
4047 \newcommand{\print@noop@unsrtglossaryunit}[2] []{%
4048   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
4049   requires the record=only or record=alsoindex package option}{}%
4050 }
```

t@getgrouptitle

```
4051 \newrobustcmd*{\@glsxtr@unsrt@getgrouptitle}[2]{%
4052   \protected@edef\@glsxtr@titlelabel{glsxtr@grouptitle@#1}%
4053   \@onelevel@sanitize\@glsxtr@titlelabel
4054   \ifcsdef{\@glsxtr@titlelabel}
4055   {\letcs{#2}{\@glsxtr@titlelabel}}%
4056   {\def#2{#1}}%
4057 }
```

\glsxtrunsrtdo Provide a user-level call to \@glsxtr@noidx@do to make it easier to define a new handler.

```
4058 \newcommand{\glsxtrunsrtdo}{\@glsxtr@noidx@do}
```

glsxtr@noidx@do Minor modification of \@gls@noidx@do to check for location field if present.

```
4059 \newcommand{\@glsxtr@noidx@do}[1]{%
4060   \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
4061   \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@location}%
4062   \ifglshasparent{#1}%
4063   {%
4064     \gls@level=\csuse{glo@\glsdetoklabel{#1}@level}\relax
4065     \ifdefvoid{\@gls@location}%
4066     {%
4067       \ifdefvoid{\@gls@loclist}%
4068       {%
4069         \subglossentry{\gls@level}{#1}{}%
4070       }%
4071       {%
4072         \subglossentry{\gls@level}{#1}%
4073         {%
4074           \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4075         }%
4076       }%
4077     }%
4078   }%
4079   \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
4080 }%
4081 }%
4082 {%
4083   \key@ifundefined{glossentry}{group}%
4084   {%
4085     \letcs{\@gls@sort}{glo@\glsdetoklabel{#1}@sort}%
4086     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
4087   }%
4088   {%
```

```

4089     \protected@xdef\@glo@thislettergrp{%
4090         \csuse{glo@\glsdetoklabel{#1}@group}}%
4091     }%
4092     \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4093     {}%
4094     {%
4095         \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
4096         \expandafter\gls@groupheading\expandafter
4097         {\@glo@thislettergrp}%
4098     }%
4099     \global\let\@gls@currentlettergroup\@glo@thislettergrp
4100     \ifdefvoid{\@gls@location}%
4101     {%
4102         \ifdefvoid{\@gls@loclist}
4103         {%
4104             \glossentry{#1}{}%
4105         }%
4106         {%
4107             \glossentry{#1}%
4108             {%
4109                 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4110             }%
4111         }%
4112     }%
4113     {%
4114         \glossentry{#1}%
4115         {%
4116             \glossaryentrynumbers{\@gls@location}%
4117         }%
4118     }%
4119 }%
4120 }

```

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.

```

4121 \@ifpackageloaded{glossaries-accsupp}
4122 {

```

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

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

```

4123     \newcommand*{\glsaccessname}[1]{%
4124         \glsnameaccessdisplay
4125         {%
4126             \glsentryname{#1}%

```

```

4127   }%
4128   {#1}%
4129   }

```

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

```

4130   \newcommand*{\Glsaccessname}[1]{%
4131     \glsnameaccessdisplay
4132     {%
4133       \Glsentryname{#1}%
4134     }%
4135     {#1}%
4136   }

```

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

```

4137   \newcommand*{\GLSaccessname}[1]{%
4138     \glsnameaccessdisplay
4139     {%
4140       \mfirstucMakeUppercase{\glsentryname{#1}}%
4141     }%
4142     {#1}%
4143   }

```

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

```

4144   \newcommand*{\glsaccesstext}[1]{%
4145     \glstextaccessdisplay
4146     {%
4147       \glsentrytext{#1}%
4148     }%
4149     {#1}%
4150   }

```

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

```

4151   \newcommand*{\Glsaccesstext}[1]{%
4152     \glstextaccessdisplay
4153     {%
4154       \Glsentrytext{#1}%
4155     }%
4156     {#1}%
4157   }

```

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

```

4158   \newcommand*{\GLSaccesstext}[1]{%
4159     \glstextaccessdisplay
4160     {%
4161       \mfirstucMakeUppercase{\glsentrytext{#1}}%
4162     }%

```



```
4163   {#1}%
4164 }
```

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

```
4165 \newcommand*\glsaccessplural}[1]{%
4166   \glspluralaccessdisplay
4167   {%
4168     \glsentryplural{#1}%
4169   }%
4170   {#1}%
4171 }
```

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

```
4172 \newcommand*\Glsaccessplural}[1]{%
4173   \glspluralaccessdisplay
4174   {%
4175     \Glsentryplural{#1}%
4176   }%
4177   {#1}%
4178 }
```

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

```
4179 \newcommand*\GLSaccessplural}[1]{%
4180   \glspluralaccessdisplay
4181   {%
4182     \mfirstucMakeUppercase{\glsentryplural{#1}}%
4183   }%
4184   {#1}%
4185 }
```

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

```
4186 \newcommand*\glsaccessfirst}[1]{%
4187   \glsfirstaccessdisplay
4188   {%
4189     \glsentryfirst{#1}%
4190   }%
4191   {#1}%
4192 }
```

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

```
4193 \newcommand*\Glsaccessfirst}[1]{%
4194   \glsfirstaccessdisplay
4195   {%
4196     \Glsentryfirst{#1}%
4197   }%
4198   {#1}%
4199 }
```

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

```
4200 \newcommand*\GLSaccessfirst}[1]{%
4201   \glsfirstaccessdisplay
4202   {%
4203     \mfirstucMakeUppercase{\glsentryfirst{#1}}%
4204   }%
4205   {#1}%
4206 }
```

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

```
4207 \newcommand*\glsaccessfirstplural}[1]{%
4208   \glsfirstpluralaccessdisplay
4209   {%
4210     \glsentryfirstplural{#1}%
4211   }%
4212   {#1}%
4213 }
```

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

```
4214 \newcommand*\GLSaccessfirstplural}[1]{%
4215   \glsfirstpluralaccessdisplay
4216   {%
4217     \Glsentryfirstplural{#1}%
4218   }%
4219   {#1}%
4220 }
```

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

```
4221 \newcommand*\GLSaccessfirstplural}[1]{%
4222   \glsfirstpluralaccessdisplay
4223   {%
4224     \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
4225   }%
4226   {#1}%
4227 }
```

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

```
4228 \newcommand*\glsaccesssymbol}[1]{%
4229   \glsymbolaccessdisplay
4230   {%
4231     \glsentrysymbol{#1}%
4232   }%
4233   {#1}%
4234 }
```

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

```

4235 \newcommand*{\Glsaccesssymbol}[1]{%
4236   \glssymbolaccessdisplay
4237   {%
4238     \Glsentrysymbol{#1}%
4239   }%
4240   {#1}%
4241 }

```

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

```

4242 \newcommand*{\GLSaccesssymbol}[1]{%
4243   \glssymbolaccessdisplay
4244   {%
4245     \mfirstucMakeUppercase{\Glsentrysymbol{#1}}%
4246   }%
4247   {#1}%
4248 }

```

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

```

4249 \newcommand*{\Glsaccesssymbolplural}[1]{%
4250   \glssymbolpluralaccessdisplay
4251   {%
4252     \Glsentrysymbolplural{#1}%
4253   }%
4254   {#1}%
4255 }

```

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

```

4256 \newcommand*{\GLSaccesssymbolplural}[1]{%
4257   \glssymbolpluralaccessdisplay
4258   {%
4259     \Glsentrysymbolplural{#1}%
4260   }%
4261   {#1}%
4262 }

```

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

```

4263 \newcommand*{\GLSaccesssymbolplural}[1]{%
4264   \glssymbolpluralaccessdisplay
4265   {%
4266     \mfirstucMakeUppercase{\Glsentrysymbolplural{#1}}%
4267   }%
4268   {#1}%
4269 }

```

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

```

4270 \newcommand*{\glsaccessdesc}[1]{%
4271   \glsdescriptionaccessdisplay

```

```

4272   {%
4273     \glentrydesc{#1}%
4274   }%
4275   {#1}%
4276 }

```

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

```

4277 \newcommand*\Glsaccessdesc}[1]{%
4278   \glsdescriptionaccessdisplay
4279   {%
4280     \glentrydesc{#1}%
4281   }%
4282   {#1}%
4283 }

```

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

```

4284 \newcommand*\GLSaccessdesc}[1]{%
4285   \glsdescriptionaccessdisplay
4286   {%
4287     \mfirstucMakeUppercase{\glentrydesc{#1}}%
4288   }%
4289   {#1}%
4290 }

```

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

```

4291 \newcommand*\glsaccessdescplural}[1]{%
4292   \glsdescriptionpluralaccessdisplay
4293   {%
4294     \glentrydescplural{#1}%
4295   }%
4296   {#1}%
4297 }

```

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

```

4298 \newcommand*\Glsaccessdescplural}[1]{%
4299   \glsdescriptionpluralaccessdisplay
4300   {%
4301     \glentrydescplural{#1}%
4302   }%
4303   {#1}%
4304 }

```

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

```

4305 \newcommand*\GLSaccessdescplural}[1]{%
4306   \glsdescriptionpluralaccessdisplay
4307   {%

```

```

4308     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
4309     }%
4310     {#1}%
4311   }

```

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

```

4312 \newcommand*{\glsaccessshort}[1]{%
4313   \glsshortaccessdisplay
4314   {%
4315     \glsentryshort{#1}%
4316   }%
4317   {#1}%
4318 }

```

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

```

4319 \newcommand*{\Glsaccessshort}[1]{%
4320   \glsshortaccessdisplay
4321   {%
4322     \Glsentryshort{#1}%
4323   }%
4324   {#1}%
4325 }

```

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

```

4326 \newcommand*{\GLSaccessshort}[1]{%
4327   \glsshortaccessdisplay
4328   {%
4329     \mfirstucMakeUppercase{\glsentryshort{#1}}%
4330   }%
4331   {#1}%
4332 }

```

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

```

4333 \newcommand*{\glsaccessshortpl}[1]{%
4334   \glsshortpluralaccessdisplay
4335   {%
4336     \glsentryshortpl{#1}%
4337   }%
4338   {#1}%
4339 }

```

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

```

4340 \newcommand*{\Glsaccessshortpl}[1]{%
4341   \glsshortpluralaccessdisplay
4342   {%
4343     \Glsentryshortpl{#1}%

```

```

4344 }%
4345 {#1}%
4346 }

```

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

```

4347 \newcommand*\LSaccessshortpl[1]{%
4348   \glshortpluralaccessdisplay
4349   {%
4350     \mfirstucMakeUppercase{\glentryshortpl{#1}}%
4351   }%
4352   {#1}%
4353 }

```

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

```

4354 \newcommand*\glsaccesslong[1]{%
4355   \glslongaccessdisplay{\glentrylong{#1}}{#1}%
4356 }

```

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

```

4357
4358 \newcommand*\Glsaccesslong[1]{%
4359   \glslongaccessdisplay{\Glentrylong{#1}}{#1}%
4360 }

```

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

```

4361 \newcommand*\GLSaccesslong[1]{%
4362   \glslongaccessdisplay
4363   {%
4364     \mfirstucMakeUppercase{\glentrylong{#1}}%
4365   }%
4366   {#1}%
4367 }

```

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

```

4368 \newcommand*\glsaccesslongpl[1]{%
4369   \glslongpluralaccessdisplay{\glentrylongpl{#1}}{#1}%
4370 }

```

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

```

4371
4372 \newcommand*\Glsaccesslongpl[1]{%
4373   \glslongpluralaccessdisplay{\Glentrylongpl{#1}}{#1}%
4374 }

```

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

```

4375 \newcommand*\GLSaccesslongpl[1]{%
4376   \glslongpluralaccessdisplay
4377   {%

```

```

4378     \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
4379     }%
4380     {#1}%
4381   }

```

End of if part

```

4382 }
4383 {

```

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

```

\glsaccessname  Display the name value (no link and no check for existence).
4384   \newcommand*{\glsaccessname}[1]{\glsentryname{#1}}

\Glsaccessname  Display the name value (no link and no check for existence) with the first letter converted to
upper case.
4385   \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

\GLSaccessname  Display the name value (no link and no check for existence). converted to upper case.
4386   \newcommand*{\GLSaccessname}[1]{%
4387     \protect\mfirstucMakeUppercase{\glsentryname{#1}}}

\glsaccessstext  Display the text value (no link and no check for existence).
4388   \newcommand*{\glsaccessstext}[1]{\glsentrytext{#1}}

\Glsaccessstext  Display the text value (no link and no check for existence) with the first letter converted to
upper case.
4389   \newcommand*{\Glsaccessstext}[1]{\Glsentrytext{#1}}

\GLSaccessstext  Display the text value (no link and no check for existence). converted to upper case.
4390   \newcommand*{\GLSaccessstext}[1]{%
4391     \protect\mfirstucMakeUppercase{\glsentrytext{#1}}}

glsaccessplural  Display the plural value (no link and no check for existence).
4392   \newcommand*{\glsaccessplural}[1]{\glsentryplural{#1}}

Glsaccessplural  Display the plural value (no link and no check for existence) with the first letter converted to
upper case.
4393   \newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}

GLSaccessplural  Display the plural value (no link and no check for existence). converted to upper case.
4394   \newcommand*{\GLSaccessplural}[1]{%
4395     \protect\mfirstucMakeUppercase{\glsentryplural{#1}}}

\glsaccessfirst  Display the first value (no link and no check for existence).
4396   \newcommand*{\glsaccessfirst}[1]{\glsentryfirst{#1}}

```

`\Glsaccessfirst` Display the first value (no link and no check for existence) with the first letter converted to upper case.
4397 `\newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}`

`\GLSaccessfirst` Display the first value (no link and no check for existence). converted to upper case.
4398 `\newcommand*{\GLSaccessfirst}[1]{%`
4399 `\protect\mfirstucMakeUppercase{\glsentryfirst{#1}}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence).
4400 `\newcommand*{\glsaccessfirstplural}[1]{\glsentryfirstplural{#1}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.
4401 `\newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.
4402 `\newcommand*{\GLSaccessfirstplural}[1]{%`
4403 `\protect\mfirstucMakeUppercase{\glsentryfirstplural{#1}}}`

`glsaccesssymbol` Display the symbol value (no link and no check for existence).
4404 `\newcommand*{\glsaccesssymbol}[1]{\glsentrysymbol{#1}}`

`GLsaccesssymbol` Display the symbol value (no link and no check for existence) with the first letter converted to upper case.
4405 `\newcommand*{\GLsaccesssymbol}[1]{\GLsentrysymbol{#1}}`

`GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.
4406 `\newcommand*{\GLSaccesssymbol}[1]{%`
4407 `\protect\mfirstucMakeUppercase{\glsentrysymbol{#1}}}`

`essymbolplural` Display the symbolplural value (no link and no check for existence).
4408 `\newcommand*{\glsaccesssymbolplural}[1]{\glsentrysymbolplural{#1}}`

`essymbolplural` Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.
4409 `\newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}`

`essymbolplural` Display the symbolplural value (no link and no check for existence). converted to upper case.
4410 `\newcommand*{\GLSaccesssymbolplural}[1]{%`
4411 `\protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}`

`\glsaccessdesc` Display the desc value (no link and no check for existence).
4412 `\newcommand*{\glsaccessdesc}[1]{\glsentrydesc{#1}}`

`\Glsaccessdesc` Display the desc value (no link and no check for existence) with the first letter converted to upper case.
4413 `\newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence). converted to upper case.
4414 `\newcommand*{\GLSaccessdesc}[1]{%`
4415 `\protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}`

`ccessdescplural` Display the descplural value (no link and no check for existence).
4416 `\newcommand*{\glsaccessdescplural}[1]{\glsentrydescplural{#1}}`

`ccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.
4417 `\newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}`

`ccessdescplural` Display the descplural value (no link and no check for existence). converted to upper case.
4418 `\newcommand*{\GLSaccessdescplural}[1]{%`
4419 `\protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}`

`\glsaccessshort` Display the short form (no link and no check for existence).
4420 `\newcommand*{\glsaccessshort}[1]{\glsentryshort{#1}}`

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).
4421 `\newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}`

`\GLSaccessshort` Display the short value (no link and no check for existence). converted to upper case.
4422 `\newcommand*{\GLSaccessshort}[1]{%`
4423 `\protect\mfirstucMakeUppercase{\glsentryshort{#1}}}`

`lsaccessshortpl` Display the short plural form (no link and no check for existence).
4424 `\newcommand*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}`

`lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).
4425 `\newcommand*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}`

`LSaccessshortpl` Display the shortplural value (no link and no check for existence). converted to upper case.
4426 `\newcommand*{\GLSaccessshortpl}[1]{%`
4427 `\protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}`

`\glsaccesslong` Display the long form (no link and no check for existence).
4428 `\newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}`

`\Glsaccesslong` Display the long form (no link and no check for existence).
4429 `\newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}`

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

```
4430 \newcommand*\GLSaccesslong}[1]{%
4431 \protect\mfirstucMakeUppercase{\glstentrylong{#1}}}
```

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

```
4432 \newcommand*\glsaccesslongpl}[1]{\glstentrylongpl{#1}}
```

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

```
4433 \newcommand*\GLsaccesslongpl}[1]{\GLstentrylongpl{#1}}
```

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

```
4434 \newcommand*\GLSaccesslongpl}[1]{%
4435 \protect\mfirstucMakeUppercase{\glstentrylongpl{#1}}}
```

End of else part

```
4436 }
```

1.5 Categories

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

```
4437 \glsaddstoragekey{category}{general}{\glscategory}
```

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

```
4438 \newcommand*\glsifcategory}[4]{%
4439 \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
4440 }
```

Categories can have attributes.

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

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

```
4441 \newcommand*\glssetcategoryattribute}[3]{%
4442 \csdef{@glsxtr@categoryattr@@#1@#2}{#3}%
4443 }
```

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

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

```
4444 \newcommand*\glsgetcategoryattribute}[2]{%
4445 \csuse{@glsxtr@categoryattr@@#1@#2}%
4446 }
```

categoryattribute

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

Tests if the category has the given attribute set.

```
4447 \newcommand*{\glshascategoryattribute}[4]{%
4448   \ifcvoid{@glstr@categoryattr@#1@#2}{#4}{#3}%
4449 }
```

\glissetattribute

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

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

```
4450 \newcommand*{\glissetattribute}[3]{%
4451   \glsetcategoryattribute{\glscategory{#1}}{#2}{#3}%
4452 }
```

\glsggetattribute

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

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

```
4453 \newcommand*{\glsggetattribute}[2]{%
4454   \glsgcategoryattribute{\glscategory{#1}}{#2}%
4455 }
```

\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.

```
4456 \newcommand*{\glshasattribute}[4]{%
4457   \ifglentryexists{#1}%
4458   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
4459   {#4}%
4460 }
```

categoryattribute

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

True if category has the attribute with the given value.

```
4461 \newcommand{\glisifcategoryattribute}[5]{%
```

```

4462 \ifcsundef{@glxtr@categoryattr@@#1@#2}%
4463 {#5}%
4464 {\ifcsstring{@glxtr@categoryattr@@#1@#2}{#3}{#4}{#5}}%
4465 }

```

`\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.

```

4466 \newcommand{\glsifattribute}[5]{%
4467   \ifglentryexists{#1}%
4468   {\glsifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
4469   {#5}%
4470 }

```

Set attributes for the default general category:

```
4471 \glssetcategoryattribute{general}{regular}{true}
```

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

```
4472 \glssetcategoryattribute{acronym}{regular}{true}
```

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

```

4473 \newcommand*{\glssetregularcategory}[1]{%
4474   \glssetcategoryattribute{#1}{regular}{true}%
4475 }

```

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

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

```

4476 \newcommand{\glsifregularcategory}[3]{%
4477   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
4478 }

```

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

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

```

4479 \newcommand{\glsifnotregularcategory}[3]{%
4480   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
4481 }

```

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

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

```
4482 \newcommand{\glsifregular}[3]{%
4483   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
4484 }
```

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

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

```
4485 \newcommand{\glsifnotregular}[3]{%
4486   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
4487 }
```

`\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.

```
4488 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
4489   \forallglossaries[#1]{#3}%
4490   {%
4491     \forlentries[#3]{#4}%
4492     {%
4493       \glsifcategory{#4}{#2}{#5}{}%
4494     }%
4495   }%
4496 }
```

`\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.

```

4497 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
4498   \forallglossaries[#1]{#4}%
4499   {%
4500     \forglsentries[#4]{#5}%
4501     {%
4502       \glsifattribute{#5}{#2}{#3}{#6}{}%
4503     }%
4504   }%
4505 }

```

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

```

4506 \ifdef\newterm
4507 {%

```

`\newterm`

```

4508   \renewcommand*\newterm}[2][ ]{%
4509     \newglossaryentry{#2}%
4510     {type={index},category=index,name={#2},%
4511      description={\glsxtrpostdescription\nopostdesc},#1}%
4512   }

```

Indexed terms are regular by default.

```

4513   \glssetcategoryattribute{index}{regular}{true}

```

`trpostdescindex`

```

4514   \newcommand*\glsxtrpostdescindex{}
4515 }
4516 {}

```

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`.

```

4517 \ifdef\printsymbols
4518 {%

```

`glsxtrnewsymbol`

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

```

4519   \newcommand*\glsxtrnewsymbol}[3][ ]{%
4520     \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
4521   }

```

Symbols are regular by default.

```

4522   \glssetcategoryattribute{symbol}{regular}{true}

```

`rpostdescsymbol`

```

4523   \newcommand*\glsxtrpostdescsymbol{}

```

```
4524 }
4525 {}
```

Similar for the numbers option.

```
4526 \ifdef\printnumbers
4527 {}%
```

glsxtrnewnumber

```
4528 \ifdef\printnumbers
4529 \newcommand*\glsxtrnewnumber[3][{}]{%
4530 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
4531 }
```

Numbers are regular by default.

```
4532 \glssetcategoryattribute{number}{regular}{true}
```

rpostdescnumber

```
4533 \newcommand*\glsxtrpostdescnumber{}
4534 }
4535 {}
```

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

```
4536 \newcommand*\glsxtrsetcategory[2]{%
4537 \@for\@glsxtr@label:=#1\do
4538 {%
4539 \glsfieldxdef{\@glsxtr@label}{category}{#2}%
4540 }%
4541 }
```

glsxtrsetcategoryforall 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.

```
4542 \newcommand*\glsxtrsetcategoryforall[2]{%
4543 \forallglossaries[#1]{\@glsxtr@type}{%
4544 \forglentries[\@glsxtr@type]{\@glsxtr@label}%
4545 {%
4546 \glsfieldxdef{\@glsxtr@label}{category}{#2}%
4547 }%
4548 }%
4549 }
```

glsxtrfieldtitlecase

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

Apply title casing to the contents of the given field.

```
4550 \newcommand*\glsxtrfieldtitlecase[2]{%
```

```

4551 \expandafter\glxtrfieldtitlecasecs\expandafter
4552   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
4553 }

```

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

```

4554 \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.

```

4555 \ifpackageloaded{glossaries-accsupp}
4556 {
4557   \renewcommand*{\glossentrydesc}[1]{%
4558     \glsdoifexistsorwarn{#1}%
4559     {%
4560       \glssetabbrvfmt{\glscategory{#1}}%

```

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

```

4561   \glsattribute{#1}{glossdescfont}%
4562   {%
4563     \edef\@glxtr@attrval{\glsattribute{#1}{glossdescfont}}%
4564     \ifcsdef{\@glxtr@attrval}%
4565     {%
4566       \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
4567     }%
4568     {%
4569       \GlossariesExtraWarning{Unknown control sequence name
4570         ‘\@glxtr@attrval’ supplied in glossdescfont attribute
4571         for entry ‘#1’. Ignoring}%
4572       \let\@glxtr@glossdescfont\@firstofone
4573     }%
4574   }%
4575   {\let\@glxtr@glossdescfont\@firstofone}%
4576   \glsifattribute{#1}{glossdesc}{firstuc}%
4577   {%
4578     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
4579   }%
4580   {%
4581     \glsifattribute{#1}{glossdesc}{title}%
4582     {%
4583       \@glxtr@do@titlecaps@warn
4584       \glsdescriptionaccessdisplay
4585     }%
4586     \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
4587   }%
4588   {#1}%

```



```

4589     }%
4590     {%
4591         \@glxtr@glossdescfont{\glsaccessdesc{#1}}%
4592     }%
4593 }%
4594 }%
4595 }
4596 }
4597 {
4598 \renewcommand*{\glossentrydesc}[1]{%
4599     \glsdoifexistsorwarn{#1}%
4600     {%
4601         \glssetabbrvfmt{\glscategory{#1}}%
4602         \glsattribute{#1}{glossdescfont}%
4603         {%
4604             \edef\@glxtr@attrval{\glsattribute{#1}{glossdescfont}}%
4605             \ifcsdef{\@glxtr@attrval}%
4606             {%
4607                 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
4608             }%
4609             {%
4610                 \GlossariesExtraWarning{Unknown control sequence name
4611                     '\@glxtr@attrval' supplied in glossdescfont attribute
4612                     for entry '#1'. Ignoring}%
4613                 \let\@glxtr@glossdescfont\@firstofone
4614             }%
4615         }%
4616         {\let\@glxtr@glossdescfont\@firstofone}%
4617         \glsattribute{#1}{glossdesc}{firstuc}%
4618         {%
4619             \@glxtr@glossdescfont{\Glsentrydesc{#1}}%
4620         }%
4621         {%
4622             \glsattribute{#1}{glossdesc}{title}%
4623             {%
4624                 \@glxtr@do@titlecaps@warn
4625                 \@glxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
4626             }%
4627             {%
4628                 \@glxtr@glossdescfont{\glsentrydesc{#1}}%
4629             }%
4630         }%
4631     }%
4632 }
4633 }

```

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

```
4634 \@ifpackageloaded{glossaries-accsupp}
```

```

4635 {
4636 \renewcommand*{\glossentryname}[1]{%
4637   \@glsdoifexistsorwarn{#1}%
4638   {%
4639     \glssetabbrvfmt{\glscategory{#1}}%

```

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

```

4640   \glsattribute{#1}{glossnamefont}%
4641   {%
4642     \edef\@glsxtr@attrval{\glsattribute{#1}{glossnamefont}}%
4643     \ifcsdef{\@glsxtr@attrval}%
4644     {%
4645       \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
4646     }%
4647     {%
4648       \GlossariesExtraWarning{Unknown control sequence name
4649         ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
4650         for entry ‘#1’. Reverting to default \string\glsnamefont}%
4651       \let\@glsxtr@glossnamefont\glsnamefont
4652     }%
4653   }%
4654   {\let\@glsxtr@glossnamefont\glsnamefont}%
4655   \glsifattribute{#1}{glossname}{firstuc}%
4656   {%
4657     \glsnameaccessdisplay
4658     {%
4659       \@glsxtr@glossnamefont{\Glsentryname{#1}}%
4660     }%
4661     {#1}%
4662   }%
4663   {%
4664     \glsifattribute{#1}{glossname}{title}%
4665     {%
4666       \@glsxtr@do@titlecaps@warn
4667       \glsnameaccessdisplay
4668       {%
4669         \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
4670       }%
4671       {#1}%
4672     }%
4673     {%
4674       \glsifattribute{#1}{glossname}{uc}%
4675       {%
4676         \glsnameaccessdisplay
4677         {%

```

Hide the label from the upper-casing command.

```

4678         \letcs{\glo@name}{glo\glsdetoklabel{#1}@name}%
4679         \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
4680       }%

```

```

4681         {#1}%
4682     }%
4683     {%
4684         \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
4685         \glsnameaccessdisplay
4686     {%
4687         \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
4688     }%
4689     {#1}%
4690 }%
4691 }%
4692 }%

```

Do post-name hook:

```

4693     \glsxtrpostnamehook{#1}%
4694 }%
4695 }
4696 }
4697 {
4698 \renewcommand*{\glossentryname}[1]{%
4699     \@glsdoifexistsorwarn{#1}%
4700     {%
4701         \glssetabbrvfmt{\glscategory{#1}}%
4702         \glsifattribute{#1}{glossnamefont}%
4703     {%
4704         \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
4705         \ifcsdef{\@glsxtr@attrval}%
4706     {%
4707         \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
4708     }%
4709     {%
4710         \GlossariesExtraWarning{Unknown control sequence name
4711         ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
4712         for entry ‘#1’. Reverting to default \string\glsnamefont}%
4713         \let\@glsxtr@glossnamefont\glsnamefont
4714     }%
4715 }%
4716 {\let\@glsxtr@glossnamefont\glsnamefont}%
4717 \glsifattribute{#1}{glossname}{firstuc}%
4718 {%
4719     \@glsxtr@glossnamefont{\Glsentryname{#1}}%
4720 }%
4721 {%
4722     \glsifattribute{#1}{glossname}{title}%
4723     {%
4724         \@glsxtr@do@titlecaps@warn
4725         \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
4726     }%
4727     {%
4728         \glsifattribute{#1}{glossname}{uc}%

```

```
4729         {%
```

Hide the label from the upper-casing command.

```
4730         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
```

```
4731         \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
```

```
4732         }%
```

```
4733         {%
```

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.

```
4734         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
```

```
4735         \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
```

```
4736         }%
```

```
4737     }%
```

```
4738 }
```

Do post-name hook.

```
4739     \glxtrpostnamehook{#1}%
```

```
4740 }%
```

```
4741 }
```

```
4742 }
```

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

```
4743 \@ifpackageloaded{glossaries-accsupp}
```

```
4744 {
```

```
4745   \renewcommand*{\Glossentryname}[1]{%
```

```
4746     \@glstdoifexistsorwarn{#1}%
```

```
4747     {%
```

```
4748       \glissetabbrvfmt{\glscategory{#1}}%
```

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

```
4749     \glshasattribute{#1}{glossnamefont}%
```

```
4750     {%
```

```
4751       \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
```

```
4752       \ifcsdef{\@glxtr@attrval}%
```

```
4753       {%
```

```
4754         \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
```

```
4755         }%
```

```
4756         {%
```

```
4757           \GlossariesExtraWarning{Unknown control sequence name
```

```
4758             ‘\@glxtr@attrval’ supplied in glossnamefont attribute
```

```
4759             for entry ‘#1’. Reverting to default \string\glsnamefont}%
```

```
4760           \let\@glxtr@glossnamefont\glsnamefont
```

```
4761           }%
```

```
4762         }%
```

```
4763       {\let\@glxtr@glossnamefont\glsnamefont}%
```

```
4764       \glsnameaccessdisplay
```

```
4765       {%
```

```
4766         \@glxtr@glossnamefont{\Glsentryname{#1}}%
```

```
4767         }%
```

```
4768       {#1}%
```

Do post-name hook:

```
4769     \glsxtrpostnamehook{#1}%
4770     }%
4771   }
4772 }
4773 {
4774   \renewcommand*{\Glossentryname}[1]{%
4775     \glsdoifexistsorwarn{#1}%
4776     {%
4777       \glssetabbrvfmt{\glscategory{#1}}%
4778       \glsattribute{#1}{glossnamefont}%
4779       {%
4780         \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
4781         \ifcsdef{\@glsxtr@attrval}%
4782         {%
4783           \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
4784           }%
4785           {%
4786             \GlossariesExtraWarning{Unknown control sequence name
4787               ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
4788               for entry ‘#1’. Reverting to default \string\glsnamefont}%
4789             \let\@glsxtr@glossnamefont\glsnamefont
4790             }%
4791           }%
4792           {\let\@glsxtr@glossnamefont\glsnamefont}%
4793           \@glsxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```
4794     \glsxtrpostnamehook{#1}%
4795     }%
4796   }
4797 }
```

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

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

```
4798 \newcommand*{\glsxtrpostnamehook}[1]{%
4799   \let\@glsnumberformat\@glsxtr@defaultnumberformat
4800   \glsxtrdoautoindexname{#1}{indexname}%

```

Allow categories to hook in here.

```
4801   \csuse{glsxtrpostname\glscategory{#1}}%
4802 }
```

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

```
4803 \newif\if@glsxtr@format@override
4804 \@glsxtr@format@overridefalse

```

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

xFormatOverride

```

4805 \ifpackageloaded{hyperref}
4806 {
    If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
    don't add it.
4807   \ifHy@hyperindex
4808     \newcommand*\GlsXtrEnableIndexFormatOverride}{%
4809       \@glsxtr@format@overridetrue
4810       \appto\theindex{\let\glshypernumber\@firstofone}%
4811     }
4812   \else
4813     \newcommand*\GlsXtrEnableIndexFormatOverride}{%
4814       \@glsxtr@format@overridetrue
4815       \appto\theindex{\let\glshypernumber\hyperpage}%
4816     }
4817   \fi
4818 }
4819 {
4820   \newcommand*\GlsXtrEnableIndexFormatOverride}{%
4821     \@glsxtr@format@overridetrue
4822   }
4823 }
4824 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

doautoindexname

```

4825 \newcommand*\glsxtrdoautoindexname}[2]{%
4826   \glsattribute{#1}{#2}%
4827   {%
    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.
4828     \@glsxtr@autoindex@setname{#1}%
    If the attribute value is simply "true" don't add an encap, otherwise use the value as the encap.
4829     \protected@edef\@glsxtr@attrval{\glsgetattribute{#1}{#2}}%
4830     \if@glsxtr@format@override
4831       \ifx\@glsnumberformat\@glsxtr@defaultnumberformat
4832       \else
4833         \let\@glsxtr@attrval\@glsnumberformat
4834       \fi
4835     \fi
4836     \ifdefstring{\@glsxtr@attrval}{true}%
4837     {}%
4838     {\eappto\@glo@name{\@glsxtr@autoindex@encap\@glsxtr@attrval}}%
4839     \expandafter\glsxtrautoindex\expandafter{\@glo@name}%

```

```

4840 }%
4841 {}%
4842 }

```

glsxtrautoindex

```
4843 \newcommand*{\glsxtrautoindex}{\index}
```

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

```

4844 \newcommand*{\@glsxtr@autoindex@setname}[1]{%
4845 \protected@edef\@glo@name{\glsxtrautoindexentry{#1}}%
4846 \glsxtrautoindexassignsort{\@glo@sort}{#1}%
4847 \@gls@checkmkidxchars\@glo@sort
4848 \@glsxtr@autoindex@doextra@esc\@glo@sort
4849 \epreto\@glo@name{\@glo@sort\@glsxtr@autoindex@at}%
4850 }

```

autoindexentry Command used for the actual part when auto-indexing.

```
4851 \newcommand*{\glsxtrautoindexentry}[1]{\string\glsentryname{#1}}
```

trautoindexsort Used to assign the sort value when auto-indexing.

```

4852 \newcommand*{\glsxtrautoindexassignsort}[2]{%
4853 \glsletentryfield{#1}{#2}{sort}%
4854 }

```

dex@doextra@esc

```
4855 \newcommand*{\@glsxtr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```

4856 \ifx\@glsxtr@autoindex@esc\@gls@quotechar
4857 \else
4858 \def\@gls@checkedmkidx{}%
4859 \edef\@@glsxtr@checkspch{%
4860 \noexpand\@glsxtr@autoindex@escquote\expandonce{#1}%
4861 \noexpand\@empty\@glsxtr@autoindex@esc\noexpand\@nnil
4862 \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4863 \@@glsxtr@checkspch
4864 \let#1\@gls@checkedmkidx\relax
4865 \fi

```

Escape actual character unless it has already been escaped.

```

4866 \ifx\@glsxtr@autoindex@at\@gls@actualchar
4867 \else
4868 \def\@gls@checkedmkidx{}%
4869 \edef\@@glsxtr@checkspch{%
4870 \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
4871 \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
4872 \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4873 \@@glsxtr@checkspch
4874 \let#1\@gls@checkedmkidx\relax
4875 \fi

```

Escape level character unless it has already been escaped.

```
4876 \ifx\@glsxtr@autoindex@level\@gls@levelchar
4877 \else
4878 \def\@gls@checkedmkidx{}%
4879 \edef\@glsxtr@checkspch{%
4880 \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
4881 \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
4882 \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4883 \@glsxtr@checkspch
4884 \let#1\@gls@checkedmkidx\relax
4885 \fi
```

Escape encap character unless it has already been escaped.

```
4886 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
4887 \else
4888 \def\@gls@checkedmkidx{}%
4889 \edef\@glsxtr@checkspch{%
4890 \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
4891 \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
4892 \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
4893 \@glsxtr@checkspch
4894 \let#1\@gls@checkedmkidx\relax
4895 \fi
4896 }
```

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`.

```
4897 \newcommand*{\@glsxtr@autoindex@at}{}
```

`trSetActualChar` Set the actual character.

```
4898 \newcommand*{\GlsXtrSetActualChar}[1]{%
4899 \gdef\@glsxtr@autoindex@at{#1}%
4900 \def\@glsxtr@autoindex@escat##1##2##3\@glsxtr@endescspch{%
4901 \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escat}{##1}{##2}{##3}%
4902 }%
4903 }
4904 \@onlypreamble\GlsXtrSetActualChar
4905 \makeatother
4906 \GlsXtrSetActualChar{@}
4907 \makeatletter
```

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

```
4908 \newcommand*{\@glsxtr@autoindex@encap}{}
```

`XtrSetEncapChar` Set the encap character.

```
4909 \newcommand*{\GlsXtrSetEncapChar}[1]{%
4910 \gdef\@glsxtr@autoindex@encap{#1}%
```



```

4911 \def\@glsxtr@autoindex@escencap##1#1##2#1##3\@glsxtr@endescspch{%
4912   \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escencap}{##1}{##2}{##3}%
4913 }%
4914 }
4915 \GlsXtrSetEncapChar{||}
4916 \@onlypreamble\GlsXtrSetEncapChar

```

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

```
4917 \newcommand*\@glsxtr@autoindex@level{}
```

`XtrSetLevelChar` Set the encap character.

```

4918 \newcommand*\GlsXtrSetLevelChar}[1]{%
4919   \gdef\@glsxtr@autoindex@level{#1}%
4920   \def\@glsxtr@autoindex@esclevel##1#1##2#1##3\@glsxtr@endescspch{%
4921     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@esclevel}{##1}{##2}{##3}%
4922   }%
4923 }
4924 \GlsXtrSetLevelChar{!}
4925 \@onlypreamble\GlsXtrSetLevelChar

```

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

```
4926 \newcommand*\@glsxtr@autoindex@esc{"}
```

`lsXtrSetEscChar` Set the escape character.

```

4927 \newcommand*\GlsXtrSetEscChar}[1]{%
4928   \gdef\@glsxtr@autoindex@esc{#1}%
4929   \def\@glsxtr@autoindex@escquote##1#1##2#1##3\@glsxtr@endescspch{%
4930     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escquote}{##1}{##2}{##3}%
4931   }%
4932 }
4933 \GlsXtrSetEscChar{"}
4934 \@onlypreamble\GlsXtrSetEscChar

```

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

```

4935 \ifdef\actualchar
4936   {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
4937 {}

```

Quote character `\quotechar`:

```

4938 \ifdef\quotechar
4939   {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
4940 {}

```

Level character `\levelchar`:

```

4941 \ifdef\levelchar
4942   {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
4943 {}

```

Encap character \encapchar:

```
4944 \ifdef\encapchar
4945 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
4946 {}}
```

leto@endescspch

```
4947 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}}
```

toindex@esc@spch

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

```
4948 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
4949 \@glxtr@tmpb=\expandafter{\@glxtr@checkedmkidx}%
4950 \toks@={#3}%
4951 \ifx\@nnil#3\relax
4952 \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
4953 \else
4954 \ifx\@nnil#4\relax
4955 \edef\@glxtr@checkedmkidx{\the\@glxtr@tmpb\the\toks@}%
4956 \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
4957 #4#5\@glxtr@endescspch}%
4958 \else
4959 \edef\@glxtr@checkedmkidx{\the\@glxtr@tmpb\the\toks@
4960 \@glxtr@autoindex@esc#1}%
4961 \def\@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
4962 \fi
4963 \fi
4964 \@glxtr@checkspch
4965 }
```

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

```
4966 \renewcommand*{\Glossentrydesc}[1]{%
4967 \glxtr@ifexistsorwarn{#1}%
4968 {%
4969 \glxtr@setabbrvfmt{\glxtr@category{#1}}%
4970 \glxtr@accessdesc{#1}%
4971 }%
4972 }
```

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

```
4973 \renewcommand*{\Glossentrysymbol}[1]{%
4974 \glxtr@ifexistsorwarn{#1}%
4975 {%
4976 \glxtr@setabbrvfmt{\glxtr@category{#1}}%
4977 \glxtr@accesssymbol{#1}%
4978 }%
4979 }
```

lossentrysymbol Redefine to set the abbreviation format and accessibility support.

```
4980 \renewcommand*{\Glossentrysymbol}[1]{%
4981   \glsdoifexistsorwarn{#1}%
4982   {%
4983     \glssetabbrvfmt{\glscategory{#1}}%
4984     \Glsaccesssymbol{#1}%
4985   }%
4986 }
```

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

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

```
4987 \newcommand*{\GlsXtrEnableInitialTagging}{%
4988   \@ifstar\s@glsextr@enabletagging\@glsextr@enabletagging
4989 }
4990 \@onlypreamble\GlsXtrEnableInitialTagging
```

r@enabletagging Starred version undefines command.

```
4991 \newcommand*{\s@glsextr@enabletagging}[2]{%
4992   \undef#2%
4993   \@glsextr@enabletagging{#1}{#2}%
4994 }
```

r@enabletagging Internal command.

```
4995 \newcommand*{\@glsextr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
4996   \@for\@glsextr@cat:=#1\do
4997   {%
4998     \ifdefempty\@glsextr@cat
4999     {}%
5000     {\glssetcategoryattribute{\@glsextr@cat}{tagging}{true}}%
5001   }%
5002   \newrobustcmd*#2[1]{##1}%
5003   \def\@glsextr@taggingcs{#2}%
5004   \renewcommand*\@glsextr@activate@initialtagging{%
5005     \let#2\@glsextr@tag
5006   }%
5007   \ifundef\@gls@preglossaryhook
5008   {\GlossariesExtraWarning{Initial tagging requires at least
5009     glossaries.sty v4.19 to work correctly}}%
5010   {}%
5011 }
```

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`

```
5012 \ifundef\mfu@checkword@do
5013 {
5014   \newcommand*\mfu@checkword@do}[1]{%
5015     \ifdefstring{\mfu@checkword@arg}{#1}%
5016     {%
5017       \let\@mfu@domakefirstuc\@firstofone
5018       \listbreak
5019     }%
5020   }%
5021 }
```

`\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.

```
5022 \ifundef\mfu@checkword
5023 {
5024   \newcommand{\@glsxtr@do@titlecaps@warn}{%
5025     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
5026       support not available}%

```

One warning should suffice.

```
5027   \let\@glsxtr@do@titlecaps@warn\relax
5028 }
5029 }
5030 {
5031   \renewcommand*\mfu@checkword}[1]{%
5032     \def\mfu@checkword@arg{#1}%
5033     \let\@mfu@domakefirstuc\makefirstuc
5034     \forlistloop\mfu@checkword@do\@mfu@nocaplist
5035   }
5036 }
5037 }
5038 {}% no patch required
```

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

```
5039 \newcommand*\@glsxtr@do@titlecaps@warn{-}
```

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

```
5040 \newcommand*\@glsxtr@activate@initialtagging{-}
```

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

```
5041 \newrobustcmd*\@glsxtr@tag}[1]{%
5042   \glsifattribute{\glscurrententrylabel}{tagging}{true}%
5043   {\glsxtrtagfont{#1}}{#1}%
5044 }
```

`\glsxtrtagfont` Used in the glossary.

```
5045 \newcommand*\glsxtrtagfont}[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.

```
5046 \ifdef \@gls@preglossaryhook
5047 {
5048   \renewcommand*{\@gls@preglossaryhook}{%
5049     \@glsxtr@activate@initialtagging
```

Since the glossaries are automatically scoped, `\@glsxtr@org@postdescription` shouldn't already be defined, but check anyway just as a precautionary measure.

```
5050   \ifundef \@glsxtr@org@postdescription
5051     {%
5052       \let \@glsxtr@org@postdescription \glspostdescription
5053       \renewcommand*{\glspostdescription}{%
5054         \ifglsentryexists{\glscurrententrylabel}%
5055           {%
5056             \glsxtrpostdescription
5057             \@glsxtr@org@postdescription
5058           }%
5059         }%
5060       }%
5061     }%
5062   }%
```

Enable the options used by `\@glsxtrp`:

```
5063   \glossxtrsetpopts
5064 }%
5065 }
5066 {}
```

`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.

```
5067 \newcommand*{\glsxtrpostdescription}{%
5068   \csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%
5069 }
```

`postdescgeneral`

```
5070 \newcommand*{\glsxtrpostdescgeneral}{}
```

`xtrpostdescterm`

```
5071 \newcommand*{\glsxtrpostdescterm}{}
```

`postdescacronym`

```
5072 \newcommand*{\glsxtrpostdescacronym}{}
```

descabbreviation

```
5073 \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.

```
5074 \renewcommand*\glspostlinkhook{%
5075   \ifglentryexists{\glslabel}{\glstrpostlinkhook}{}%
5076 }
```

trpostlinkhook

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

```
5077 \newcommand*\glstrpostlinkhook{%
5078   \glstrdiscardperiod{\glslabel}%
5079   {\glstrpostlinkendsentence}%
5080   {\glstrpostlink}%
5081 }
```

\glstrpostlink

```
5082 \newcommand*\glstrpostlink{%
5083   \csuse{glstrpostlink\glscategory{\glslabel}}%
5084 }
```

linkendsentence

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

```
5085 \newcommand*\glstrpostlinkendsentence{%
5086   \ifcsdef{glstrpostlink\glscategory{\glslabel}}
5087   {%
5088     \csuse{glstrpostlink\glscategory{\glslabel}}%
5089     .\spacefactor\sfcode`. \relax
5090   }%
5091   {%
```

Put the full stop back.

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

```
5092   \spacefactor\sfcode`. \relax
5093   }%
5094 }
```

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.

```
5095 \newcommand*\glstrpostlinkAddDescOnFirstUse{%
5096   \glstrifwasfirstuse{\space(\glsaccessdesc{\glslabel})}{}%
5097 }
```

ymbolOnFirstUse

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

```

5098 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
5099   \glxtrifwasfirstuse
5100   {%
5101     \ifglshassymbol{\glslabel}{\space(\glssaccesssymbol{\glslabel})}{}%
5102   }%
5103   {}%
5104 }

```

`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).

```

5105 \newcommand*{\glxtrdiscardperiod}[3]{%
5106   \glxtrifwasfirstuse
5107   {%
5108     \glusifattribute{#1}{retainfirstuseperiod}{true}%
5109     {#3}%
5110     {%
5111       \glusifattribute{#1}{discardperiod}{true}%
5112       {%
5113         \glusifplural
5114         {%
5115           \glusifattribute{#1}{pluraldiscardperiod}{true}%
5116           {\glxtrifperiod{#2}{#3}}%
5117           {#3}%
5118         }%
5119         {%
5120           \glxtrifperiod{#2}{#3}%
5121           }%
5122         }%
5123         {#3}%
5124       }%
5125     }%
5126     {%
5127       \glusifattribute{#1}{discardperiod}{true}%
5128       {%
5129         \glusifplural
5130         {%
5131           \glusifattribute{#1}{pluraldiscardperiod}{true}%
5132           {\glxtrifperiod{#2}{#3}}%
5133           {#3}%
5134         }%
5135         {%
5136           \glxtrifperiod{#2}{#3}%
5137           }%
5138         }%
5139         {#3}%
5140       }%
5141     }

```

`\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`

```
5142 \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 ' ').

```
5143 \newcommand*\glxtr@punclist}{.,;?!}
```

`punctuationmark` Add character to punctuation list.

```
5144 \newcommand*\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}
```

`unctuationmarks` Reset the punctuation list.

```
5145 \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`.)

```
5146 \newcommand*\glxtrifnextpunc}[2]{%
5147 \def\reserved@a{#1}%
5148 \def\reserved@b{#2}%
5149 \futurelet\@glspunc@token\glxtr@ifnextpunc
5150 }
```

`\glxtr@ifnextpunc`

```
5151 \newcommand*\glxtr@ifnextpunc}{%
5152 \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
5153 \reserved@b
5154 }
```

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

```
5155 \newcommand*\glxtr@ifpunctoken}[1]{%
5156 \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
5157 }
```

`\glxtr@ifpunctoken`

```
5158 \def\@glxtr@ifpunctoken#1#2{%
5159 \let\reserved@d=#2%
5160 \ifx\reserved@d\@nnil
5161 \let\glxtr@next\@glxtr@notfoundinlist
5162 \else
5163 \ifx#1\reserved@d
5164 \let\glxtr@next\@glxtr@foundinlist
5165 \else
```



```

5166     \let\glsxtr@next\@glsxtr@ifpunctoken
5167     \fi
5168     \fi
5169     \glsxtr@next#1%
5170 }

```

xtr@foundinlist

```

5171 \def\@glsxtr@foundinlist#1\@nnil{\@firstoftwo}

```

@notfoundinlist

```

5172 \def\@glsxtr@notfoundinlist#1{\@secondoftwo}

```

glsxtrdopostpunc

```
\glsxtrdopostpunc{<code>}
```

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

```

5173 \newcommand{\glsxtrdopostpunc}[1]{%
5174   \glsxtrifnextpunc{\@glsxtr@swaptwo{#1}}{#1}%
5175 }

```

@glsxtr@swaptwo

```

5176 \newcommand{\@glsxtr@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.

```

5177 \define@key{glsxtrabbrv}{category}{%
5178   \edef\glscategorylabel{#1}%
5179   \ifcsdef{@glsabbrv@current@#1}%
5180   {%

```

Warning should already have been issued.

```

5181   \let\@glsxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
5182   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
5183   \glsxtr@applyabbrvstyle{\csname @glsabbrv@current@#1\endcsname}%
5184   \let\GlsXtrWarnDeprecatedAbbrStyle\@glsxtr@orgwarndep
5185 }%
5186 {}%
5187 }

```

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

```
5188 \define@key{glstrabbrv}{shortplural}{%
5189   \def\@gls@shortpl{#1}%
5190 }
```

Similarly for the long plural form.

```
5191 \define@key{glstrabbrv}{longplural}{%
5192   \def\@gls@longpl{#1}%
5193 }
```

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

`\glsshortpltok`

```
5194 \newtoks\glsshortpltok
```

`\glslongpltok`

```
5195 \newtoks\glslongpltok
```

`xtr@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.

```
5196 \newcommand*{\@glstr@insertdots}[2]{%
5197   \def#1{}}
5198   \@glstr@insert@dots#1#2\@nnil
5199 }
```

`xtr@insert@dots`

```
5200 \newcommand*{\@glstr@insert@dots}[2]{%
5201   \ifx\@nnil#2\relax
5202   \let\@glstr@insert@dots@next\@gobble
5203   \else
5204   \ifx\relax#2\relax
5205   \else
5206     \appto#1{#2.}%
5207   \fi
5208   \let\@glstr@insert@dots@next\@glstr@insert@dots
5209   \fi
5210   \@glstr@insert@dots@next#1%
5211 }
```

Similarly provide a way of replacing spaces with `\glstrwordsep`, which first needs to be defined:

`\glstrwordsep`

```
5212 \newcommand*{\glstrwordsep}{\space}
```

Each word is marked with

`\glxtrword`

```
5213 \newcommand*{\glxtrword}[1]{#1}
```

`tr@markwordseps`

```
5214 \newcommand*{\@glxtr@markwordseps}[2]{%
5215   \def#1{}}%
5216   \@glxtr@mark@wordseps#1#2 \@nnil
5217 }
```

`r@mark@wordseps`

```
5218 \def\@glxtr@mark@wordseps#1#2 #3{%
5219   \ifdefempty{#1}%
5220   {\def#1{\protect\glxtrword{#2}}}%
5221   {\appto#1{\protect\glxtrwordsep\protect\glxtrword{#2}}}%
5222   \ifx\@nnil#3\relax
5223   \let\@glxtr@mark@wordseps@next\relax
5224   \else
5225   \def\@glxtr@mark@wordseps@next{%
5226     \@glxtr@mark@wordseps#1#3}%
5227   \fi
5228   \@glxtr@mark@wordseps@next
5229 }
```

`newabbreviation` Define a new generic abbreviation.

```
5230 \newcommand*{\newabbreviation}[4][]{%
5231   \glxtr@newabbreviation{#1}{#2}{#3}{#4}%
5232 }
```

`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.)

```
5233 \newcommand*{\glxtr@newabbreviation}[4]{%
5234   \glskeylisttok{#1}%
5235   \glslabeltok{#2}%
5236   \glsshorttok{#3}%
5237   \glslongtok{#4}%
```

Save the original short and long values (before attribute settings modify them).

```
5238   \def\glxtrorgshort{#3}%
5239   \def\glxtrorglong{#4}%
```

Get the category.

```
5240   \def\glscategorylabel{abbreviation}%
5241   \glxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
```

Ignore the shortplural and longplural keys.

```
5242   \setkeys*{glxtrabbrv}[shortplural,longplural]{#1}%
```

Set the default long plural

```
5243 \def\@gls@longpl{#4\glspluralsuffix}%  
5244 \let\@gls@default@longpl\@gls@longpl
```

Has the markwords attribute been set?

```
5245 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%  
5246 {%  
5247   \@glsxtr@markwordseps\@gls@long{#4}%  
5248   \expandafter\def\expandafter\@gls@longpl\expandafter  
5249     {\@gls@long\glspluralsuffix}%  
5250   \let\@gls@default@longpl\@gls@longpl
```

Update \glslongtok.

```
5251   \expandafter\glslongtok\expandafter{\@gls@long}%  
5252 }%  
5253 {}%
```

Has the markshortwords attribute been set? (Not compatible with insertdots.)

```
5254 \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%  
5255 {%  
5256   \@glsxtr@markwordseps\@gls@short{#3}%  
5257 }%  
5258 {}%
```

Has the insertdots attribute been set?

```
5259   \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%  
5260   {%  
5261     \@glsxtr@insertdots\@gls@short{#3}%  
5262     \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%  
5263   }%  
5264   {\def\@gls@short{#3}}%  
5265 }%
```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```
5266 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%  
5267 {%  
5268   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
5269     '\abbrvpluralsuffix}%  
5270 }%  
5271 {}%
```

Has the noshortplural attribute been set?

```
5272   \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%  
5273   {%  
5274     \let\@gls@shortpl\@gls@short  
5275   }%  
5276   {%  
5277     \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
5278       \abbrvpluralsuffix}%  
5279   }%  
5280 }%
```

Update \glsshorttok:

```
5281 \expandafter\glsshorttok\expandafter{\@gls@short}%
```

Hook for further customisation if required:

```
5282 \glxtrnewabbrevpresetkeyhook{#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).

```
5283 \setkeys*{glxtrabbrv}[category]{#1}%
```

Has the plural been explicitly set?

```
5284 \ifx\@gls@default@longpl\@gls@longpl
```

```
5285 \else
```

Has the markwords attribute been set?

```
5286 \glsifcategoryattribute{\gls@categorylabel}{markwords}{true}%
```

```
5287 {%
```

```
5288 \expandafter\@glxtr@markwordseps\expandafter\@gls@longpl\expandafter
```

```
5289 {\@gls@longpl}%
```

```
5290 }%
```

```
5291 {}%
```

```
5292 \fi
```

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

```
5293 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
```

```
5294 \expandafter\glslongpltok\expandafter{\@gls@longpl}%
```

Do any extra setup provided by hook:

```
5295 \newabbreviationhook
```

Define this entry:

```
5296 \protected@edef\@do@newglossaryentry{%
```

```
5297 \noexpand\newglossaryentry{\the\glslabeltok}%
```

```
5298 {%
```

```
5299 type=\glxtrabbrvtype,%
```

```
5300 category=abbreviation,%
```

```
5301 short={\the\glsshorttok},%
```

```
5302 shortplural={\the\glsshortpltok},%
```

```
5303 long={\the\glslongtok},%
```

```
5304 longplural={\the\glslongpltok},%
```

```
5305 name={\the\glsshorttok},%
```

```
5306 \CustomAbbreviationFields,%
```

```
5307 \the\glskeylisttok
```

```
5308 }%
```

```
5309 }%
```

```
5310 \@do@newglossaryentry
```

```
5311 \GlsXtrPostNewAbbreviation
```

```
5312 }
```

evpresetkeyhook Hook for extra stuff in \newabbreviation

```
5313 \newcommand*{\glxtrnewabbrevpresetkeyhook}[3]{}
```

`NewAbbreviation` Hook used by abbreviation styles.
5314 `\newcommand*{\GlsXtrPostNewAbbreviation}{}`

`abbreviationhook` Hook for use with `\newabbreviation`.
5315 `\newcommand*{\newabbreviationhook}{}`

`abbreviationFields`
5316 `\newcommand*{\CustomAbbreviationFields}{}`

`\glsxtrparen` For the parenthetical styles.
5317 `\newcommand*{\glsxtrparen}[1]{(#1)}`

`glsxtrfullformat` Full format without case change.
5318 `\newcommand*{\glsxtrfullformat}[2]{%`
5319 `\glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%`
5320 `\glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%`
5321 `}`

`glsxtrfullformat` Full format with case change.
5322 `\newcommand*{\Glsxtrfullformat}[2]{%`
5323 `\glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%`
5324 `\glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%`
5325 `}`

`glsxtrfullplformat` Plural full format without case change.
5326 `\newcommand*{\glsxtrfullplformat}[2]{%`
5327 `\glsfirstlongfont{\glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%`
5328 `\glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}}}%`
5329 `}`

`glsxtrfullplformat` Plural full format with case change.
5330 `\newcommand*{\Glsxtrfullplformat}[2]{%`
5331 `\glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%`
5332 `\glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}}}%`
5333 `}`

`\glsxtrfullsep` Separator used by full format is a space by default. The argument is the entry's label.
5334 `\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).

`glsxtrinfullformat` Full format without case change.
5335 `\newcommand*{\glsxtrinfullformat}{\glsxtrfullformat}`

`glsxtrinfullformat` Full format with case change.
5336 `\newcommand*{\Glsxtrinfullformat}{\Glsxtrfullformat}`

`xtrfullplformat` Plural full format without case change.

```
5337 \newcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}
```

`inefullplformat` Plural full format with case change.

```
5338 \newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}
```

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

`\glentryfull`

```
5339 \renewcommand*{\glentryfull}[1]{\glxtrinlinefullformat{#1}{}}
```

`\Glsentryfull`

```
5340 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\glentryfullpl`

```
5341 \renewcommand*{\glentryfullpl}[1]{\glxtrinlinefullplformat{#1}{}}
```

`\Glsentryfullpl`

```
5342 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`sfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
5343 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}
```

`abbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.

```
5344 \newcommand*{\glabbrvdefaultfont}[1]{\glabbrvfont{#1}}
```

`\glabbrvfont` Font changing command used for the abbreviation on subsequent use.

```
5345 \newcommand*{\glabbrvfont}[1]{\glabbrvdefaultfont{#1}}
```

`abbrvdefaultfont`

```
5346 \newcommand*{\glabbrvdefaultfont}[1]{#1}
```

`\glslongfont` Font changing command used for the long form in commands like `\glxtrlong`.

```
5347 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}
```

`longdefaultfont` Default font changing command used for the long form in commands like `\glxtrlong`.

```
5348 \newcommand*{\glslongdefaultfont}[1]{#1}
```

`lsfirstlongfont` Font changing command used for the long form on first use or in the full format.

```
5349 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}
```

`longdefaultfont`

```
5350 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}
```

brvpluralsuffix Default plural suffix. Allow an alternative default suffix for abbreviations.

```
5351 \newcommand*{\glxtrabbrvpluralsuffix}{\glspluralsuffix}
```

brvpluralsuffix Default plural suffix.

```
5352 \newcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}
```

\glxtrfull Full form (no case-change).

```
5353 \newrobustcmd*{\glxtrfull}{\@gls@hyp@opt\ns@glxtrfull}
```

```
5354 \newcommand*\ns@glxtrfull[2][\]{%
```

```
5355 \new@ifnextchar[{\@glxtr@full{#1}{#2}}{%
```

```
5356 \glxtr@full{#1}{#2}[]}%
```

```
5357 }
```

\@glxtr@full Low-level macro:

```
5358 \def\@glxtr@full#1#2[#3]{%
```

```
5359 \glsdoifexists{#2}%
```

```
5360 {%
```

```
5361 \glssetabbrvfmt{\glscategory{#2}}%
```

```
5362 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
5363 \let\glsifplural\@secondoftwo
```

```
5364 \let\glsapscase\@firstofthree
```

```
5365 \let\glsinsert\@empty
```

```
5366 \def\glscustomtext{\glsxtrinlinefullformat{#2}{#3}}%
```

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

```
5367 \glsxtrsetupfulldefs
```

```
5368 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
```

```
5369 }%
```

```
5370 \glspostlinkhook
```

```
5371 }
```

trsetupfulldefs

```
5372 \newcommand*{\glsxtrsetupfulldefs}{%
```

```
5373 \let\glsxtrifwasfirstuse\@firstoftwo
```

```
5374 }
```

\Glsxtrfull Full form (first letter uppercase).

```
5375 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\ns@Glsxtrfull}
```

```
5376 \newcommand*\ns@Glsxtrfull[2][\]{%
```

```
5377 \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}{%
```

```
5378 \@Glsxtr@full{#1}{#2}[]}%
```

```
5379 }
```

\@Glsxtr@full Low-level macro:

```
5380 \def\@Glsxtr@full#1#2[#3]{%
```



```

5381 \glsdoifexists{#2}%
5382 {%
5383   \glssetabbrvfmt{\glscategory{#2}}%
5384   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5385   \let\glsifplural\@secondoftwo
5386   \let\glscapscase\@secondofthree
5387   \let\glsinsert\@empty
5388   \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
5389   \glsxtrsetupfulldefs
5390   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5391 }%
5392 \glspostlinkhook
5393 }

```

`\GLSxtrfull` Full form (all uppercase).

```

5394 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\ns@GLSxtrfull}
5395 \newcommand*\ns@GLSxtrfull[2] [] {%
5396   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}%
5397     {\@GLSxtr@full{#1}{#2} []}%
5398 }

```

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

```

5399 \def\@GLSxtr@full#1#2[#3] {%
5400   \glsdoifexists{#2}%
5401   {%
5402     \glssetabbrvfmt{\glscategory{#2}}%
5403     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5404     \let\glsifplural\@secondoftwo
5405     \let\glscapscase\@thirdofthree
5406     \let\glsinsert\@empty
5407     \def\glscustomtext{\mfirstucMakeUppercase{\glsxtrinlinefullformat{#2}{#3}}}%
5408     \glsxtrsetupfulldefs
5409     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5410   }%
5411   \glspostlinkhook
5412 }

```

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

```

5413 \newrobustcmd*{\glsxtrfullpl}{\@gls@hyp@opt\ns@glsxtrfullpl}
5414 \newcommand*\ns@glsxtrfullpl[2] [] {%
5415   \new@ifnextchar[{\@glsxtr@fullpl{#1}{#2}}%
5416     {\@glsxtr@fullpl{#1}{#2} []}%
5417 }

```

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

```

5418 \def\@glsxtr@fullpl#1#2[#3] {%
5419   \glsdoifexists{#2}%
5420   {%
5421     \glssetabbrvfmt{\glscategory{#2}}%

```

```

5422 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5423 \let\glsifplural\@firstoftwo
5424 \let\gls caps case\@firstofthree
5425 \let\glsinsert\@empty
5426 \def\gls custom text{\gls xtr inline full pl format{#2}{#3}}%
5427 \gls xtr setup full defs
5428 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5429 }%
5430 \gls post link hook
5431 }

```

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

```

5432 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
5433 \newcommand*\ns@Glsxtrfullpl[2] []{%
5434 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
5435 {\@Glsxtr@fullpl{#1}{#2} []}%
5436 }

```

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

```

5437 \def\@Glsxtr@fullpl#1#2[#3]{%
5438 \glsdoifexists{#2}%
5439 {%
5440 \glssetabbrvfmt{\gls category{#2}}%
5441 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5442 \let\glsifplural\@firstoftwo
5443 \let\gls caps case\@secondofthree
5444 \let\glsinsert\@empty
5445 \def\gls custom text{\Glsxtr inline full pl format{#2}{#3}}%
5446 \gls xtr setup full defs
5447 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5448 }%
5449 \gls post link hook
5450 }

```

`\GLSxtrfullpl` Plural full form (all upper case).

```

5451 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\@ns@GLSxtrfullpl}
5452 \newcommand*\ns@GLSxtrfullpl[2] []{%
5453 \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}%
5454 {\@GLSxtr@fullpl{#1}{#2} []}%
5455 }

```

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

```

5456 \def\@GLSxtr@fullpl#1#2[#3]{%
5457 \glsdoifexists{#2}%
5458 {%
5459 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5460 \let\glsifplural\@firstoftwo
5461 \let\gls caps case\@thirdofthree
5462 \let\glsinsert\@empty

```

```

5463   \def\glscustomtext{%
5464     \mfirstucMakeUppercase{\glxtrinlinefullplformat{#2}{#3}}}%
5465   \glxtrsetupfulldefs
5466   \@gls@link[#1]{#2}{\cename gls@\glstype @entryfmt\endcename}%
5467 }%
5468   \glspostlinkhook
5469 }

```

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

`\glxtrshort`

```

5470 \newrobustcmd*{\glxtrshort}{\@gls@hyp@opt\ns@glxtrshort}

```

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

```

5471 \newcommand*{\ns@glxtrshort}[2] [] {%
5472   \new@ifnextchar[{\@glxtrshort{#1}{#2}}{\@glxtrshort{#1}{#2} [] }%
5473 }

```

Read in the final optional argument:

```

5474 \def\@glxtrshort#1#2[#3] {%
5475   \glsdoifexists{#2}%
5476   {%

```

Need to make sure `\glsabbrvfont` is set correctly.

```

5477   \glssetabbrvfmt{\glscategory{#2}}%
5478   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5479   \let\glxtrifwasfirstuse\@secondoftwo
5480   \let\glsifplural\@secondoftwo
5481   \let\glscapscase\@firstofthree
5482   \let\glsinsert\@empty
5483   \def\glscustomtext{%
5484     \glsabbrvfont{\glsaccessshort{#2}\ifglxtrinsertinside#3\fi}%
5485     \ifglxtrinsertinside\else#3\fi
5486   }%
5487   \@gls@link[#1]{#2}{\cename gls@\glstype @entryfmt\endcename}%
5488 }%
5489   \glspostlinkhook
5490 }

```

`\Glsxtrshort`

```

5491 \newrobustcmd*{\Glsxtrshort}{\@gls@hyp@opt\ns@Glsxtrshort}

```

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

```

5492 \newcommand*{\ns@Glsxtrshort}[2] [] {%
5493   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} [] }%
5494 }

```

Read in the final optional argument:

```

5495 \def\@Glsxtrshort#1#2[#3] {%
5496   \glsdoifexists{#2}%
5497   {%

```

```

5498 \glssetabbrvfmt{\glscategory{#2}}%
5499 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5500 \let\glsxtrifwasfirstuse\@secondoftwo
5501 \let\glsifplural\@secondoftwo
5502 \let\glscapscase\@secondofthree
5503 \let\glsinsert\@empty
5504 \def\glscustomtext{%
5505 \glsabbrvfont{\Glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
5506 \ifglsxtrininsertinside\else#3\fi
5507 }%
5508 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5509 }%
5510 \glspostlinkhook
5511 }

```

\GLSxtrshort

```

5512 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}

```

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

```

5513 \newcommand*{\ns@GLSxtrshort}[2] [] {%
5514 \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} []}]%
5515 }

```

Read in the final optional argument:

```

5516 \def\@GLSxtrshort#1#2[#3] {%
5517 \glsdoifexists{#2}%
5518 {%
5519 \glssetabbrvfmt{\glscategory{#2}}%
5520 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5521 \let\glsxtrifwasfirstuse\@secondoftwo
5522 \let\glsifplural\@secondoftwo
5523 \let\glscapscase\@thirdofthree
5524 \let\glsinsert\@empty
5525 \def\glscustomtext{%
5526 \mfirstucMakeUppercase
5527 {\glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
5528 \ifglsxtrininsertinside\else#3\fi
5529 }%
5530 }%
5531 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5532 }%
5533 \glspostlinkhook
5534 }

```

\glsxtrlong

```

5535 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}

```

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

```

5536 \newcommand*{\ns@glsxtrlong}[2] [] {%
5537 \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2} []}]%
5538 }

```

Read in the final optional argument:

```
5539 \def\@glsxtrlong#1#2[#3]{%
5540   \glsdoifexists{#2}%
5541   {%
5542     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5543     \let\glsxtrifwasfirstuse\@secondoftwo
5544     \let\glsifplural\@secondoftwo
5545     \let\glscapscase\@firstofthree
5546     \let\glsinsert\@empty
5547     \def\glscustomtext{%
5548       \glslongfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
5549       \ifglsxtrininsertinside\else#3\fi
5550     }%
5551     \@gls@link[#1]{#2}{\cename gls@\glstype @entryfmt\endcename}%
5552   }%
5553   \glspostlinkhook
5554 }
```

\Glsxtrlong

```
5555 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\ns@Glsxtrlong}
   Define the un-starred form. Need to determine if there is a final optional argument
5556 \newcommand*{\ns@Glsxtrlong}[2] [] {%
5557   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
5558 }
```

Read in the final optional argument:

```
5559 \def\@Glsxtrlong#1#2[#3]{%
5560   \glsdoifexists{#2}%
5561   {%
5562     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5563     \let\glsxtrifwasfirstuse\@secondoftwo
5564     \let\glsifplural\@secondoftwo
5565     \let\glsapsaps\@secondofthree
5566     \let\glsinsert\@empty
5567     \def\glscustomtext{%
5568       \glslongfont{\Glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
5569       \ifglsxtrininsertinside\else#3\fi
5570     }%
5571     \@gls@link[#1]{#2}{\cename gls@\glstype @entryfmt\endcename}%
5572   }%
5573   \glspostlinkhook
5574 }
```

\GLSxtrlong

```
5575 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}
   Define the un-starred form. Need to determine if there is a final optional argument
5576 \newcommand*{\ns@GLSxtrlong}[2] [] {%
5577   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
5578 }
```

Read in the final optional argument:

```
5579 \def\@GLSxtrlong#1#2[#3]{%
5580   \glsdoifexists{#2}%
5581   {%
5582     \let\do@glS@link@checkfirsthyper\@glS@link@nocheckfirsthyper
5583     \let\glSxtrifwasfirstuse\@secondoftwo
5584     \let\glSifplural\@secondoftwo
5585     \let\glScapscase\@thirdofthree
5586     \let\glSinsert\@empty
5587     \def\glScustomtext{%
5588       \mfirstucMakeUppercase
5589       {\glSlongfont{\glSaccesslong{#2}\ifglSxtrininsertinside#3\fi}%
5590       \ifglSxtrininsertinside\else#3\fi
5591     }%
5592   }%
5593   \@glS@link[#1]{#2}{\csname glS@\glStype @entryfmt\endcsname}%
5594 }%
5595 \glSpostlinkhook
5596 }
```

Plural short forms:

`\glSxtrshortpl`

```
5597 \newrobustcmd*{\glSxtrshortpl}{\@glS@hyp@opt\ns@glSxtrshortpl}
   Define the un-starred form. Need to determine if there is a final optional argument
5598 \newcommand*{\ns@glSxtrshortpl}[2] []{%
5599   \new@ifnextchar[{\@glSxtrshortpl{#1}{#2}}{\@glSxtrshortpl{#1}{#2} []}%
5600 }
```

Read in the final optional argument:

```
5601 \def\@glSxtrshortpl#1#2[#3]{%
5602   \glsdoifexists{#2}%
5603   {%
5604     \glSsetabbrvfmt{\glScategory{#2}}%
5605     \let\do@glS@link@checkfirsthyper\@glS@link@nocheckfirsthyper
5606     \let\glSxtrifwasfirstuse\@secondoftwo
5607     \let\glSifplural\@firstoftwo
5608     \let\glScapscase\@firstofthree
5609     \let\glSinsert\@empty
5610     \def\glScustomtext{%
5611       \glSabbrvfont{\glSaccessshortpl{#2}\ifglSxtrininsertinside#3\fi}%
5612       \ifglSxtrininsertinside\else#3\fi
5613     }%
5614     \@glS@link[#1]{#2}{\csname glS@\glStype @entryfmt\endcsname}%
5615   }%
5616   \glSpostlinkhook
5617 }
```

`\Glsxtrshortpl`

```
5618 \newrobustcmd*{\Glsxtrshortpl}{\@glS@hyp@opt\ns@Glsxtrshortpl}
```

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

```
5619 \newcommand*\ns@GLSxtrshortpl}[2] [] {%
5620   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}]%
5621 }
```

Read in the final optional argument:

```
5622 \def\@GLSxtrshortpl#1#2[#3] {%
5623   \glsdoifexists{#2}%
5624   {%
5625     \glssetabbrvfmt{\glscategory{#2}}%
5626     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5627     \let\glsxtrifwasfirstuse\@secondoftwo
5628     \let\glsifplural\@firstoftwo
5629     \let\glsapscase\@secondofthree
5630     \let\glsinsert\@empty
5631     \def\glscustomtext{%
5632       \glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
5633       \ifglsxtrininsertinside\else#3\fi
5634     }%
5635     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5636   }%
5637   \glspostlinkhook
5638 }
```

\GLSxtrshortpl

```
5639 \newrobustcmd*\GLSxtrshortpl}{\@gls@hyp@opt\ns@GLSxtrshortpl}
```

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

```
5640 \newcommand*\ns@GLSxtrshortpl}[2] [] {%
5641   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}]%
5642 }
```

Read in the final optional argument:

```
5643 \def\@GLSxtrshortpl#1#2[#3] {%
5644   \glsdoifexists{#2}%
5645   {%
5646     \glssetabbrvfmt{\glscategory{#2}}%
5647     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5648     \let\glsxtrifwasfirstuse\@secondoftwo
5649     \let\glsifplural\@firstoftwo
5650     \let\glsapscase\@thirdofthree
5651     \let\glsinsert\@empty
5652     \def\glscustomtext{%
5653       \mfirstucMakeUppercase
5654       {\glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
5655       \ifglsxtrininsertinside\else#3\fi
5656     }%
5657   }%
5658   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5659 }
```

```
5660 \glspostlinkhook
5661 }
```

Plural long forms:

`\glsxtrlongpl`

```
5662 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\@ns@glsxtrlongpl}
Define the un-starred form. Need to determine if there is a final optional argument
5663 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
5664 \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
5665 }
Read in the final optional argument:
5666 \def\@glsxtrlongpl#1#2[#3] {%
5667 \glsdoifexists{#2}%
5668 {%
5669 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5670 \let\glsxtrifwasfirstuse\@secondoftwo
5671 \let\glsifplural\@firstoftwo
5672 \let\gls caps case\@firstofthree
5673 \let\glsinsert\@empty
5674 \def\gls custom text{%
5675 \gls long font{\gls access long pl{#2}\ifglsxtrininsertinside#3\fi}%
5676 \ifglsxtrininsertinside\else#3\fi
5677 }%
5678 \@gls@link[#1]{#2}{\csname gls@\gls type @entryfmt\endcsname}%
5679 }%
5680 \gls post link hook
5681 }
```

`\Glsxtrlongpl`

```
5682 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}
Define the un-starred form. Need to determine if there is a final optional argument
5683 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
5684 \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
5685 }
Read in the final optional argument:
5686 \def\@Glsxtrlongpl#1#2[#3] {%
5687 \glsdoifexists{#2}%
5688 {%
5689 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5690 \let\glsxtrifwasfirstuse\@secondoftwo
5691 \let\glsifplural\@firstoftwo
5692 \let\gls caps case\@secondofthree
5693 \let\glsinsert\@empty
5694 \def\gls custom text{%
5695 \gls long font{\Gls access long pl{#2}\ifglsxtrininsertinside#3\fi}%
5696 \ifglsxtrininsertinside\else#3\fi
```



```

5697 }%
5698 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5699 }%
5700 \glspostlinkhook
5701 }

```

\GLSxtrlongpl

```

5702 \newrobustcmd*{\GLSxtrlongpl}{\@gls@hyp@opt\@ns@GLSxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
5703 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
5704   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
5705 }

    Read in the final optional argument:
5706 \def\@GLSxtrlongpl#1#2[#3]{%
5707   \glsdoifexists{#2}%
5708   {%
5709     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5710     \let\glsxtrifwasfirstuse\@secondoftwo
5711     \let\glsifplural\@firstoftwo
5712     \let\glscapscase\@thirdofthree
5713     \let\glsinsert\@empty
5714     \def\glscustomtext{%
5715       \mfirstucMakeUppercase
5716       {\glsfont{\glsaccesslongpl{#2}\ifglsxtrinsetinside#3\fi}%
5717       \ifglsxtrinsetinside\else#3\fi
5718     }%
5719   }%
5720   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5721 }%
5722 \glspostlinkhook
5723 }

```

\glssetabbrvfmt Set the current format for the given category (or the abbreviation category if unset).

```

5724 \newcommand*{\glssetabbrvfmt}[1]{%
5725   \ifcsdef{@glsabbrv@current@#1}%
5726   {\glsxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
5727   {\glsxtr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
5728 }

```

sxtrgenabbrvfmt Similar to \glsngenacfmt, but for abbreviations.

```

5729 \newcommand*{\glsxtrgenabbrvfmt}{%
5730   \ifdefempty\glscustomtext
5731   {%
5732     \ifglsused\glslabel
5733     {%

```

Subsequent use:

5734 `\glsifplural`
5735 `{%`

Subsequent plural form:

5736 `\glscapscase`
5737 `{%`

Subsequent plural form, don't adjust case:

5738 `\glxtrsubsequentplfmt{\glslabel}{\glsinsert}%`
5739 `}%`
5740 `{%`

Subsequent plural form, make first letter upper case:

5741 `\Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%`
5742 `}%`
5743 `{%`

Subsequent plural form, all caps:

5744 `\mfirstucMakeUppercase`
5745 `{\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%`
5746 `}%`
5747 `}%`
5748 `{%`

Subsequent singular form

5749 `\glscapscase`
5750 `{%`

Subsequent singular form, don't adjust case:

5751 `\glxtrsubsequentfmt{\glslabel}{\glsinsert}%`
5752 `}%`
5753 `{%`

Subsequent singular form, make first letter upper case:

5754 `\Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%`
5755 `}%`
5756 `{%`

Subsequent singular form, all caps:

5757 `\mfirstucMakeUppercase`
5758 `{\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%`
5759 `}%`
5760 `}%`
5761 `}%`
5762 `{%`

First use:

5763 `\glsifplural`
5764 `{%`

First use plural form:

5765 `\glscapscase`
5766 `{%`

First use plural form, don't adjust case:

```
5767      \glxtrfullplformat{\glslabel}{\glsinsert}%  
5768      }%  
5769      {%
```

First use plural form, make first letter upper case:

```
5770      \Glsxtrfullplformat{\glslabel}{\glsinsert}%  
5771      }%  
5772      {%
```

First use plural form, all caps:

```
5773      \mfirstucMakeUppercase  
5774      {\glxtrfullplformat{\glslabel}{\glsinsert}}%  
5775      }%  
5776      }%  
5777      {%
```

First use singular form

```
5778      \glscapscase  
5779      {%
```

First use singular form, don't adjust case:

```
5780      \glxtrfullformat{\glslabel}{\glsinsert}%  
5781      }%  
5782      {%
```

First use singular form, make first letter upper case:

```
5783      \Glsxtrfullformat{\glslabel}{\glsinsert}%  
5784      }%  
5785      {%
```

First use singular form, all caps:

```
5786      \mfirstucMakeUppercase  
5787      {\glxtrfullformat{\glslabel}{\glsinsert}}%  
5788      }%  
5789      }%  
5790      }%  
5791      }%  
5792      {%
```

User supplied text.

```
5793      \glscustomtext  
5794      }%  
5795      }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
5796 \newcommand*{\glxtrsubsequentfmt}[2]{%  
5797   \glsabbrvfont{\glsaccessshort{#1}}\ifglxtrinertinside #2\fi}%  
5798   \ifglxtrinertinside \else#2\fi  
5799 }  
5800 \let\glxtrdefaultsubsequentfmt\glxtrsubsequentfmt
```

```

subsequentplfmt Subsequent use format (plural no case change).
5801 \newcommand*\glxtrsubsequentplfmt}[2]{%
5802 \glsabbrvfont{\glsaccessshortpl{#1}\ifglxtrinsertinside #2\fi}%
5803 \ifglxtrinsertinside \else#2\fi
5804 }
5805 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt

```

```

trsubsequentfmt Subsequent use format (singular, first letter uppercase).
5806 \newcommand*\Glsxtrsubsequentfmt}[2]{%
5807 \glsabbrvfont{\Glsaccessshortpl{#1}\ifglxtrinsertinside #2\fi}%
5808 \ifglxtrinsertinside \else#2\fi
5809 }
5810 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt

```

```

subsequentplfmt Subsequent use format (plural, first letter uppercase).
5811 \newcommand*\Glsxtrsubsequentplfmt}[2]{%
5812 \glsabbrvfont{\Glsaccessshortpl{#1}\ifglxtrinsertinside #2\fi}%
5813 \ifglxtrinsertinside \else#2\fi
5814 }
5815 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt

```

1.6.1 Abbreviation Styles Setup

```

breviationstyle
5816 \newcommand*\setabbreviationstyle}[2][abbreviation]{%
5817 \ifcsundef{@glsabbrv@dispstyle@setup@#2}
5818 {%
5819 \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
5820 }%
5821 {%

Have abbreviations already been defined for this category?
5822 \ifcsstring{@glsabbrv@current@#1}{#2}%
5823 {%

Style already set.
5824 }%
5825 {%
5826 \def\@glxtr@dostylewarn{}%
5827 \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
5828 {%
5829 \def\@glxtr@dostylewarn{\GlossariesWarning{Abbreviation
5830 style has been switched \MessageBreak
5831 for category ‘#1’, \MessageBreak
5832 but there have already been entries \MessageBreak
5833 defined for this category. Unwanted \MessageBreak
5834 side-effects may result}}%
5835 \@endfortrue
5836 }%
5837 \@glxtr@dostylewarn

```

Set up the style for the given category.

```
5838     \csdef{@glsabbrv@current@#1}{#2}%
5839     \glsxtr@applyabbrvstyle{#2}%
5840   }%
5841 }%
5842 }
```

`applyabbrvstyle` Apply the abbreviation style without existence check.

```
5843 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
5844   \csuse{@glsabbrv@dispstyle@setup@#1}%
5845   \csuse{@glsabbrv@dispstyle@fmts@#1}%
5846 }
```

`r@applyabbrvfmt` Only apply the style formats.

```
5847 \newcommand*{\glsxtr@applyabbrvfmt}[1]{%
5848   \csuse{@glsabbrv@dispstyle@fmts@#1}%
5849 }
```

`briviationstyle` 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.

```
5850 \newcommand*{\newabbreviationstyle}[3]{%
5851   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
5852   {%
5853     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
5854     defined}{}%
5855   }%
5856   {%
5857     \csdef{@glsabbrv@dispstyle@setup@#1}{%
```

Initialise hook to do nothing. The style may change this.

```
5858     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
5859     #2}%
5860   \csdef{@glsabbrv@dispstyle@fmts@#1}{%
```

Assume in-line form is the same as first use. The style may change this.

```
5861   \renewcommand*{\glsxtrinlinelinefullformat}{\glsxtrfullformat}%
5862   \renewcommand*{\Glsxtrinlinelinefullformat}{\Glsxtrfullformat}%
5863   \renewcommand*{\glsxtrinlinelinefullplformat}{\glsxtrfullplformat}%
5864   \renewcommand*{\Glsxtrinlinelinefullplformat}{\Glsxtrfullplformat}%
```

Reset `\glsxtrsubsequentfmt` etc in case a style changes this.

```
5865   \let\glsxtrsubsequentfmt\glsxtrdefaultsubsequentfmt
5866   \let\glsxtrsubsequentplfmt\glsxtrdefaultsubsequentplfmt
5867   \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
5868   \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
5869   #3}%
5870 }%
5871 }
```

re abbreviationstyle

```
5872 \newcommand*{\renewabbreviationstyle}[3]{%
5873   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
5874   {%
5875     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
5876   }%
5877   {%
5878     \csdef{@glsabbrv@dispstyle@setup@#1}{%
```

Initialise hook to do nothing. The style may change this.

```
5879     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
5880     #2}%
5881     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
```

Assume in-line form is the same as first use. The style may change this.

```
5882     \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
5883     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
5884     \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
5885     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
5886     #3}%
5887   }%
5888 }
```

re abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style’s name.

```
5889 \newcommand*{\letabbreviationstyle}[2]{%
5890   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
5891   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
5892 }
```

re deprecated@abbrstyle

```
\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}
```

Define a synonym for a deprecated abbreviation style.

```
5893 \newcommand*{@@glsxtr@deprecated@abbrstyle}[2]{%
5894   \csdef{@glsabbrv@dispstyle@setup@#1}{%
5895     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
5896   \csuse{@glsabbrv@dispstyle@setup@#2}%
5897   }%
5898   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
5899 }
```

re deprecatedAbbrStyle Generate warning for deprecated style use.

```
5900 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
5901   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
5902   use ‘#2’ instead}%
5903 }
```

eAbbrStyleSetup

```
5904 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
5905   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
5906   {%
5907     \PackageError{glossaries-extra}%
5908     {Unknown abbreviation style definitions ‘#1’}{}%
5909   }%
5910   {%
5911     \csname @glsabbrv@dispstyle@setup@#1\endcsname
5912   }%
5913 }
```

seAbbrStyleFmts

```
5914 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
5915   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
5916   {%
5917     \PackageError{glossaries-extra}%
5918     {Unknown abbreviation style formats ‘#1’}{}%
5919   }%
5920   {%
5921     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
5922   }%
5923 }
```

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.

```
5924 \newif\ifglsxtrinsertinside
5925 \glsxtrinsertinsidefalse
```

long-short

```
5926 \newabbreviationstyle{long-short}%
5927 {%
5928   \renewcommand*{\CustomAbbreviationFields}{%
5929     name={\protect\glsabbrvfont{\the\glsshorttok}},
5930     sort={\the\glsshorttok},
5931     first={\protect\glsfirstlongfont{\the\glslongtok}}%
5932     \protect\glsxtrfullsep{\the\glslabeltok}}%
```

```

5933 \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
5934 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}}%
5935 \protect\glxtrfullsep{\the\glslabeltok}}%
5936 \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
5937 plural={\protect\glsabbrvfont{\the\glsshortpltok}}},%
5938 description={\the\glslongtok}}%

```

Unset the regular attribute if it has been set.

```

5939 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5940 \glshasattribute{\the\glslabeltok}{regular}}%
5941 {%
5942 \glissetattribute{\the\glslabeltok}{regular}{false}}%
5943 }%
5944 {}%
5945 }%
5946 }%
5947 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

5948 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
5949 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
5950 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
5951 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
5952 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

5953 \renewcommand*{\glxtrfullformat}[2]{%
5954 \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
5955 \ifglxtrinsertinside\else##2\fi
5956 \glxtrfullsep{##1}}%
5957 \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}}%
5958 }%
5959 \renewcommand*{\glxtrfullplformat}[2]{%
5960 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5961 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}}%
5962 \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}}%
5963 }%
5964 \renewcommand*{\Glsxtrfullformat}[2]{%
5965 \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
5966 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}}%
5967 \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}}%
5968 }%
5969 \renewcommand*{\Glsxtrfullplformat}[2]{%
5970 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
5971 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}}%
5972 \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}}%
5973 }%
5974 }

```

Set this as the default style for general abbreviations:


```
5975 \setabbreviationstyle{long-short}
```

ngshortdescsort

```
5976 \newcommand*{\glxtrlongshortdescsort}{%
5977 \expandonce\glxtrorglong\space (\expandonce\glxtrorgshort)%
5978 }
```

ngshortdescname

```
5979 \newcommand*{\glxtrlongshortdescname}{%
5980 \protect\glsfont{\the\glsfonttok}
5981 \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
5982 }
```

long-short-desc User supplies description. The long form is included in the name.

```
5983 \newabbreviationstyle{long-short-desc}%
5984 {%
5985 \renewcommand*{\CustomAbbreviationFields}{%
5986 name={\glxtrlongshortdescname},
5987 sort={\glxtrlongshortdescsort},%
5988 first={\protect\glsfirstlongfont{\the\glsfonttok}%
5989 \protect\glxtrfullsep{\the\glslabeltok}%
5990 \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
5991 firstplural={\protect\glsfirstlongfont{\the\glsfonttok}%
5992 \protect\glxtrfullsep{\the\glslabeltok}%
5993 \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
```

The text key should only have the short form.

```
5994 text={\protect\glsabbrvfont{\the\glsshorttok}},%
5995 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
5996 }%
```

Unset the regular attribute if it has been set.

```
5997 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
5998 \glsattribute{\the\glslabeltok}{regular}%
5999 {%
6000 \glssetattribute{\the\glslabeltok}{regular}{false}%
6001 }%
6002 {}%
6003 }%
6004 }%
6005 {%
6006 \GlsXtrUseAbbrStyleFmts{long-short}%
6007 }
```

short-long Short form followed by long form in parenthesis on first use.

```
6008 \newabbreviationstyle{short-long}%
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\glsextrfullsep{\the\glslabeltok}}%
6016 \glsextrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6017 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6018 \protect\glsextrfullsep{\the\glslabeltok}}%
6019 \glsextrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6020 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

6021 \renewcommand*\GlsXtrPostNewAbbreviation}{%
6022 \glsattribute{\the\glslabeltok}{regular}}%
6023 {%
6024 \glssetattribute{\the\glslabeltok}{regular}{false}}%
6025 }%
6026 {}%
6027 }%
6028 }%
6029 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6030 \renewcommand*\abbrvpluralsuffix{\glsextrabbrvpluralsuffix}%
6031 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
6032 \renewcommand*\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6033 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6034 \renewcommand*\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6035 \renewcommand*\glsextrfullformat}[2]{%
6036 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsextrininsertinside##2\fi}%
6037 \ifglsextrininsertinside\else##2\fi
6038 \glsextrfullsep{##1}}%
6039 \glsextrparen{\glsfirstlongfont{\glsaccesslong{##1}}}}%
6040 }%
6041 \renewcommand*\glsextrfullplformat}[2]{%
6042 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsextrininsertinside##2\fi}%
6043 \ifglsextrininsertinside\else##2\fi
6044 \glsextrfullsep{##1}}%
6045 \glsextrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}}%
6046 }%
6047 \renewcommand*\Glsxtrfullformat}[2]{%
6048 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsextrininsertinside##2\fi}%
6049 \ifglsextrininsertinside\else##2\fi\glsextrfullsep{##1}}%
6050 \glsextrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}}%
6051 }%
6052 \renewcommand*\Glsxtrfullplformat}[2]{%
6053 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsextrininsertinside##2\fi}%
6054 \ifglsextrininsertinside\else##2\fi\glsextrfullsep{##1}}%
6055 \glsextrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}}%

```

```
6056 }%
6057 }
```

ortlongdescsort

```
6058 \newcommand*{\glxtrshortlongdescsort}{\the\glsshorttok}
```

ortlongdescname

```
6059 \newcommand*{\glxtrshortlongdescname}{%
6060   \protect\glsabbrvfont{\the\glsshorttok}
6061   \glxtrparen{\protect\glslongfont{\the\glslongtok}}}%
6062 }
```

short-long-desc User supplies description. The long form is included in the name.

```
6063 \newabbreviationstyle{short-long-desc}%
6064 {%
6065   \renewcommand*{\CustomAbbreviationFields}{%
6066     name={\glxtrshortlongdescname},
6067     sort={\glxtrshortlongdescsort},
6068     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6069     \protect\glxtrfullsep{\the\glslabeltok}}%
6070     \glxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6071     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6072     \protect\glxtrfullsep{\the\glslabeltok}}%
6073     \glxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6074     text={\protect\glsabbrvfont{\the\glsshorttok}},%
6075     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6076 }%
```

Unset the regular attribute if it has been set.

```
6077 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6078   \glsattribute{\the\glslabeltok}{regular}}%
6079   {%
6080     \glssetattribute{\the\glslabeltok}{regular}{false}}%
6081   }%
6082   {}%
6083 }%
6084 }%
6085 {%
6086   \GlsXtrUseAbbrStyleFmts{short-long}}%
6087 }
```

ongfootnotefont Only used by the “footnote” styles.

```
6088 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
6089 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%
```

xtrabbrvfootnote

```
\glxtrabbrvfootnote{<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`).

```
6090 \newcommand*{\glxtrabbrvfootnote}[2]{\footnote{#2}}
```

`footnote` Short form followed by long form in footnote on first use.

```
6091 \newabbreviationstyle{footnote}%
6092 {%
6093   \renewcommand*{\CustomAbbreviationFields}{%
6094     name={\protect\glsabbrvfont{\the\glsshorttok}},
6095     sort={\the\glsshorttok},
6096     description={\the\glslongtok},%
6097     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6098       \protect\glxtrabbrvfootnote{\the\glslabeltok}}%
6099     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
6100     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6101       \protect\glxtrabbrvfootnote{\the\glslabeltok}}%
6102     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
6103     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.

```
6104 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6105   \glsssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
6106   \glshasattribute{\the\glslabeltok}{regular}%
6107   {%
6108     \glsssetattribute{\the\glslabeltok}{regular}{false}%
6109   }%
6110   {}%
6111 }%
6112 }%
6113 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
6114 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
6115 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6116 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6117 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6118 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```
6119 \renewcommand*{\glxtrfullformat}[2]{%
6120   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglxtrinsetinside##2\fi}%

```

```

6121 \ifglxtrinsertinside\else##2\fi
6122 \protect\glxtrabbrvfootnote{##1}%
6123 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6124 }%
6125 \renewcommand*{\glxtrfullplformat}[2]{%
6126 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6127 \ifglxtrinsertinside\else##2\fi
6128 \protect\glxtrabbrvfootnote{##1}%
6129 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6130 }%
6131 \renewcommand*{\Glsxtrfullformat}[2]{%
6132 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6133 \ifglxtrinsertinside\else##2\fi
6134 \protect\glxtrabbrvfootnote{##1}%
6135 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6136 }%
6137 \renewcommand*{\Glsxtrfullplformat}[2]{%
6138 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6139 \ifglxtrinsertinside\else##2\fi
6140 \protect\glxtrabbrvfootnote{##1}%
6141 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6142 }%

```

The first use full form and the inline full form use the short (long) style.

```

6143 \renewcommand*{\glxtrinlinelinefullformat}[2]{%
6144 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6145 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6146 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6147 }%
6148 \renewcommand*{\glxtrinlinelinefullplformat}[2]{%
6149 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6150 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6151 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6152 }%
6153 \renewcommand*{\Glsxtrinlinelinefullformat}[2]{%
6154 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6155 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6156 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6157 }%
6158 \renewcommand*{\Glsxtrinlinelinefullplformat}[2]{%
6159 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6160 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6161 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6162 }%
6163 }

```

short-footnote

```
6164 \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.

```
6165 \newabbreviationstyle{postfootnote}%
6166 {%
6167   \renewcommand*{\CustomAbbreviationFields}{%
6168     name={\protect\glsabbrvfont{\the\glsshorttok}},
6169     sort={\the\glsshorttok},
6170     description={\the\glslongtok},%
6171     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
6172     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
6173     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.

```
6174 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6175   \csdef{glsxtrpostlink\glscategorylabel}{%
6176     \glsxtrifwasfirstuse
6177     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
6178       \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}%
6179       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
6180     }%
6181   }%
6182 }%
6183 \glsattribute{\the\glslabeltok}{regular}%
6184 {%
6185   \glssetattribute{\the\glslabeltok}{regular}{false}%
6186 }%
6187 }%
6188 }%
```

The footnote needs to be suppressed in the inline form, so `\glsxtrfull` must set the first use switch off.

```
6189 \renewcommand*{\glsxtrsetupfulldefs}{%
6190   \let\glsxtrifwasfirstuse\@secondoftwo
6191 }%
6192 }%
6193 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6194 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6195 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
6196 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
6197 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
6198 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
6199 \renewcommand*\glsxtrfullformat[2]{%
```

```

6200 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6201 \ifglxtrinsertinside\else##2\fi
6202 }%
6203 \renewcommand*{\glxtrfullplformat}[2]{%
6204 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6205 \ifglxtrinsertinside\else##2\fi
6206 }%
6207 \renewcommand*{\Glsxtrfullformat}[2]{%
6208 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6209 \ifglxtrinsertinside\else##2\fi
6210 }%
6211 \renewcommand*{\Glsxtrfullplformat}[2]{%
6212 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6213 \ifglxtrinsertinside\else##2\fi
6214 }%

```

The first use full form and the inline full form use the short (long) style.

```

6215 \renewcommand*{\glxtrinlinelinefullformat}[2]{%
6216 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6217 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6218 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6219 }%
6220 \renewcommand*{\glxtrinlinelinefullplformat}[2]{%
6221 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6222 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6223 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6224 }%
6225 \renewcommand*{\Glsxtrinlinelinefullformat}[2]{%
6226 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6227 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6228 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6229 }%
6230 \renewcommand*{\Glsxtrinlinelinefullplformat}[2]{%
6231 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6232 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6233 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6234 }%
6235 }

```

rt-postfootnote

```
6236 \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.

```

6237 \newabbreviationstyle{short}%
6238 {%
6239 \renewcommand*{\CustomAbbreviationFields}{%

```

```

6240   name={\protect\glsabbrvfont{\the\glsshorttok}},
6241   sort={\the\glsshorttok},
6242   first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
6243   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
6244   text={\protect\glsabbrvfont{\the\glsshorttok}},
6245   plural={\protect\glsabbrvfont{\the\glsshortpltok}},
6246   description={\the\glslongtok}}%
6247 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6248   \glssetattribute{\the\glslabeltok}{regular}{true}}%
6249 }%
6250 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6251 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6252 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6253 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6254 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6255 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

6256 \renewcommand*{\glsxtrinelinefullformat}[2]{%
6257   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
6258   \ifglsxtrininsertinside##2\fi}%
6259   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
6260   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}}%
6261 }%
6262 \renewcommand*{\glsxtrinelinefullplformat}[2]{%
6263   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
6264   \ifglsxtrininsertinside##2\fi}%
6265   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
6266   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}}%
6267 }%
6268 \renewcommand*{\Glsxtrinelinefullformat}[2]{%
6269   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
6270   \ifglsxtrininsertinside##2\fi}%
6271   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
6272   \glsxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}}%
6273 }%
6274 \renewcommand*{\Glsxtrinelinefullplformat}[2]{%
6275   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
6276   \ifglsxtrininsertinside##2\fi}%
6277   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
6278   \glsxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}}%
6279 }%

```

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.

```

6280 \renewcommand*{\glsxtrfullformat}[2]{%
6281   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
6282   \ifglsxtrininsertinside\else##2\fi
6283 }%

```



```

6284 \renewcommand*{\glxtrfullplformat}[2]{%
6285   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6286   \ifglxtrinsertinside\else##2\fi
6287 }%
6288 \renewcommand*{\Glsxtrfullformat}[2]{%
6289   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6290   \ifglxtrinsertinside\else##2\fi
6291 }%
6292 \renewcommand*{\Glsxtrfullplformat}[2]{%
6293   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6294   \ifglxtrinsertinside\else##2\fi
6295 }%
6296 }

```

Set this as the default style for acronyms:

```
6297 \setabbreviationstyle[acronym]{short}
```

short-nolong

```
6298 \letabbreviationstyle{short-nolong}{short}
```

short-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

6299 \newabbreviationstyle{short-nolong-noreg}%
6300 {%
6301   \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

6302 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6303   \glsattribute{\the\glslabeltok}{regular}%
6304   {%
6305     \glssetattribute{\the\glslabeltok}{regular}{false}%
6306   }%
6307   {}}%
6308 }%
6309 }%
6310 {%
6311   \GlsXtrUseAbbrStyleFmts{short-nolong}%
6312 }

```

trshortdescname

```

6313 \newcommand*{\glxtrshortdescname}{%
6314   \protect\glsabbrvfont{\the\glsshorttok}%
6315 }

```

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.

```

6316 \newabbreviationstyle{short-desc}%
6317 {%
6318   \renewcommand*{\CustomAbbreviationFields}{%
6319     name={\glxtrshortdescname},

```

```

6320     sort={\the\glsshorttok},
6321     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
6322     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
6323     text={\protect\glsabbrvfont{\the\glsshorttok}},
6324     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
6325     description={\the\glslongtok}}%
6326 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6327   \glssetattribute{\the\glslabeltok}{regular}{true}}%
6328 }%
6329 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6330 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6331 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6332 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6333 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6334 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

6335 \renewcommand*{\glsxtrinlinefullformat}[2]{%
6336   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6337   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6338   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6339 }%
6340 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
6341   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6342   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6343   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6344 }%
6345 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6346   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6347   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6348   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6349 }%
6350 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6351   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6352   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6353   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6354 }%

```

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.

```

6355 \renewcommand*{\glsxtrfullformat}[2]{%
6356   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6357   \ifglsxtrininsertinside\else##2\fi
6358 }%
6359 \renewcommand*{\glsxtrfullplformat}[2]{%
6360   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6361   \ifglsxtrininsertinside\else##2\fi
6362 }%
6363 \renewcommand*{\Glsxtrfullformat}[2]{%

```

```

6364   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6365   \ifglxtrinsertinside\else##2\fi
6366 }%
6367 \renewcommand*{\GlsXtrFullPlFormat}[2]{%
6368   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6369   \ifglxtrinsertinside\else##2\fi
6370 }%
6371 }

```

ort-nolong-desc

```
6372 \letabbreviationstyle{short-nolong-desc}{short-desc}
```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

6373 \newabbreviationstyle{short-nolong-desc-noreg}%
6374 {%
6375   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
6376   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6377     \glsattribute{\the\glslabeltok}{regular}%
6378     {%
6379       \glssetattribute{\the\glslabeltok}{regular}{false}%
6380     }%
6381   }%
6382 }%
6383 }%
6384 {%
6385   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
6386 }

```

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.

```

6387 \newabbreviationstyle{long-desc}%
6388 {%
6389   \renewcommand*{\CustomAbbreviationFields}{%
6390     name={\protect\protect\glslongfont{\the\glslongtok}},
6391     sort={\the\glslongtok},
6392     first={\protect\glsfirstlongfont{\the\glslongtok}},
6393     firstplural={\protect\glsfirstlongfont{\the\glslongtok}},
6394     text={\glslongfont{\the\glslongtok}},
6395     plural={\glslongfont{\the\glslongtok}}%
6396   }%
6397   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6398     \glssetattribute{\the\glslabeltok}{regular}{true}}%
6399 }%
6400 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6401 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
6402 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
6403 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
6404 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
6405 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

6406 \renewcommand*\glxtrsubsequentfmt[2]{%
6407   \glslongfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
6408   \ifglxtrininsertinside \else##2\fi
6409 }%
6410 \renewcommand*\glxtrsubsequentplfmt[2]{%
6411   \glslongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
6412   \ifglxtrininsertinside \else##2\fi
6413 }%
6414 \renewcommand*\Glsxtrsubsequentfmt[2]{%
6415   \glslongfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
6416   \ifglxtrininsertinside \else##2\fi
6417 }%
6418 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
6419   \glslongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
6420   \ifglxtrininsertinside \else##2\fi
6421 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

6422 \renewcommand*\glxtrininlinefullformat[2]{%
6423   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6424   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6425   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6426 }%
6427 \renewcommand*\glxtrininlinefullplformat[2]{%
6428   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6429   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6430   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6431 }%
6432 \renewcommand*\Glsxtrininlinefullformat[2]{%
6433   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6434   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6435   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6436 }%
6437 \renewcommand*\Glsxtrininlinefullplformat[2]{%
6438   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6439   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6440   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6441 }%

```

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.

```

6442 \renewcommand*\glxtrfullformat[2]{%
6443   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6444   \ifglxtrininsertinside\else##2\fi

```

```

6445 }%
6446 \renewcommand*{\Glsxtrfullplformat}[2]{%
6447   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
6448   \ifglsxtrinsertinside\else##2\fi
6449 }%
6450 \renewcommand*{\Glsxtrfullformat}[2]{%
6451   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
6452   \ifglsxtrinsertinside\else##2\fi
6453 }%
6454 \renewcommand*{\Glsxtrfullplformat}[2]{%
6455   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
6456   \ifglsxtrinsertinside\else##2\fi
6457 }%
6458 }

```

`long-noshort-desc` Provide a synonym that matches similar styles.

```
6459 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

`short-desc-noreg` Like `long-noshort-desc` but doesn't set the regular attribute.

```

6460 \newabbreviationstyle{long-noshort-desc-noreg}%
6461 {%
6462   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
6463   \GlsXtrPostNewAbbreviation{%
6464     \glsattribute{\the\glslabeltok}{regular}%
6465     {%
6466       \glssetattribute{\the\glslabeltok}{regular}{false}%
6467     }%
6468   }%
6469 }%
6470 }%
6471 {%
6472   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
6473 }

```

`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.

```

6474 \newabbreviationstyle{long}%
6475 {%
6476   \renewcommand*{\CustomAbbreviationFields}{%
6477     name={\protect\glsabbrvfont{\the\glsshorttok}},
6478     sort={\the\glsshorttok},
6479     first={\protect\glsfirstlongfont{\the\glslongtok}},
6480     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
6481     text={\glslongfont{\the\glslongtok}},
6482     plural={\glslongfont{\the\glslongpltok}},%
6483     description={\the\glslongtok}%

```

```

6484 }%
6485 \renewcommand*\GlsXtrPostNewAbbreviation}{%
6486   \glssetattribute{\the\glslabeltok}{regular}{true}}%
6487 }%
6488 {%
6489   \GlsXtrUseAbbrStyleFmts{long-desc}%
6490 }

```

`long-noshort` Provide a synonym that matches similar styles.

```
6491 \letabbreviationstyle{long-noshort}{long}
```

`g-noshort-noreg` Like `long-noshort` but doesn't set the regular attribute.

```

6492 \newabbreviationstyle{long-noshort-noreg}%
6493 {%
6494   \GlsXtrUseAbbrStyleSetup{long-noshort}%
6495   Unset the regular attribute if it has been set.
6496   \renewcommand*\GlsXtrPostNewAbbreviation}{%
6497     \glsattribute{\the\glslabeltok}{regular}%
6498     \glssetattribute{\the\glslabeltok}{regular}{false}%
6499   }%
6500   {}%
6501 }%
6502 }%
6503 {%
6504   \GlsXtrUseAbbrStyleFmts{long-noshort}%
6505 }

```

1.6.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

`\glsxtrscfont` Maintained for backward-compatibility.

```
6506 \newcommand*\glsxtrscfont[1]{\textsc{#1}}
```

`\glsabbrvscfont` Added for consistent naming.

```
6507 \newcommand*\glsabbrvscfont{\glsxtrscfont}
```

`sxtrfirstscfont` Maintained for backward-compatibility.

```
6508 \newcommand*\sxtrfirstscfont[1]{\glsabbrvscfont{#1}}
```

`firstabbrvscfont` Added for consistent naming.

```
6509 \newcommand*\firstabbrvscfont{\sxtrfirstscfont}
```

and for the default short form suffix:

`\glsxtrscsuffix`

```
6510 \newcommand*\glsxtrscsuffix{\glsstextup{\glsxtrabbrvpluralsuffix}}
```

long-short-sc

```
6511 \newabbreviationstyle{long-short-sc}%
6512 {%
6513   \renewcommand*{\CustomAbbreviationFields}{%
6514     name={\protect\glsabbrvscfont{\the\glsshorttok}},
6515     sort={\the\glsshorttok},
6516     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
6517       \protect\glsxtrfullsep{\the\glslabeltok}%
6518       \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
6519     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
6520       \protect\glsxtrfullsep{\the\glslabeltok}%
6521       \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
6522     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},%
6523     description={\the\glslongtok}}%
6524 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6525   \glsattribute{\the\glslabeltok}{regular}%
6526   {%
6527     \glssetattribute{\the\glslabeltok}{regular}{false}%
6528   }%
6529   {}%
6530 }%
6531 }%
6532 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
6533 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
6534 \renewcommand*{\glsabbrvfont[1]{\glsabbrvscfont{##1}}}%
6535 \renewcommand*{\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}}
```

Use the default long fonts.

```
6536 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6537 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
6538 \renewcommand*{\glsxtrfullformat}[2]{%
6539   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinertinside##2\fi}%
6540   \ifglsxtrinertinside\else##2\fi
6541   \glsxtrfullsep{##1}%
6542   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
6543 }%
6544 \renewcommand*{\glsxtrfullplformat}[2]{%
6545   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinertinside##2\fi}%
6546   \ifglsxtrinertinside\else##2\fi\glsxtrfullsep{##1}%
6547   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
6548 }%
6549 \renewcommand*{\Glsxtrfullformat}[2]{%
6550   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinertinside##2\fi}%
6551   \ifglsxtrinertinside\else##2\fi\glsxtrfullsep{##1}%
6552   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
6553 }%
```

```

6554 \renewcommand*{\Glsxtrfullplformat}[2]{%
6555   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinertinside##2\fi}%
6556   \ifglsxtrinertinside\else##2\fi\glsxtrfullsep{##1}%
6557   \glsxtrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
6558 }%
6559 }

```

g-short-sc-desc

```

6560 \newabbreviationstyle{long-short-sc-desc}%
6561 {%
6562   \renewcommand*{\CustomAbbreviationFields}{%
6563     name={\glsxtrlongshortdescname},
6564     sort={\glsxtrlongshortdescsort},%
6565     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
6566       \protect\glsxtrfullsep{\the\glslabeltok}%
6567       \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
6568     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
6569       \protect\glsxtrfullsep{\the\glslabeltok}%
6570       \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
6571     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
6572     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
6573   }%

```

Unset the regular attribute if it has been set.

```

6574 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6575   \glsattribute{\the\glslabeltok}{regular}%
6576   {%
6577     \glssetattribute{\the\glslabeltok}{regular}{false}%
6578   }%
6579   {}%
6580 }%
6581 }%
6582 {%

```

As long-short-sc style:

```

6583 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
6584 }

```

Now the short (long) version

```

6585 \newabbreviationstyle{short-sc-long}%
6586 {%
6587   \renewcommand*{\CustomAbbreviationFields}{%
6588     name={\protect\glsabbrvscfont{\the\glsshorttok}},
6589     sort={\the\glsshorttok},
6590     description={\the\glslongtok},%
6591     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
6592       \protect\glsxtrfullsep{\the\glslabeltok}%
6593       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
6594     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
6595       \protect\glsxtrfullsep{\the\glslabeltok}%

```



```

6596     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
6597     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

6598 \renewcommand*\GlsXtrPostNewAbbreviation}{%
6599   \glsattribute{\the\glslabeltok}{regular}%
6600   {%
6601     \glssetattribute{\the\glslabeltok}{regular}{false}%
6602   }%
6603   {}%
6604 }%
6605 }%
6606 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

6607 \renewcommand*\abbrvpluralsuffix{\protect\glxtrscsuffix}%
6608 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
6609 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
6610 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
6611 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6612 \renewcommand*\glxtrfullformat}[2]{%
6613   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6614   \ifglxtrininsertinside\else##2\fi
6615   \glxtrfullsep{##1}%
6616   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
6617 }%
6618 \renewcommand*\glxtrfullplformat}[2]{%
6619   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6620   \ifglxtrininsertinside\else##2\fi
6621   \glxtrfullsep{##1}%
6622   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
6623 }%
6624 \renewcommand*\Glsxtrfullformat}[2]{%
6625   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6626   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6627   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
6628 }%
6629 \renewcommand*\Glsxtrfullplformat}[2]{%
6630   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6631   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6632   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
6633 }%
6634 }

```

As before but user provides description

```

6635 \newabbreviationstyle{short-sc-long-desc}%
6636 {%
6637   \renewcommand*\CustomAbbreviationFields}{%

```

```

6638   name={\glxtrshortlongdescname},
6639   sort={\glxtrshortlongdescsort},
6640   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
6641     \protect\glxtrfullsep{\the\glslabeltok}%
6642     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
6643   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
6644     \protect\glxtrfullsep{\the\glslabeltok}%
6645     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
6646   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
6647   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
6648 }%

```

Unset the regular attribute if it has been set.

```

6649 \renewcommand*\GlsXtrPostNewAbbreviation{%
6650   \glsattribute{\the\glslabeltok}{regular}%
6651   {%
6652     \glsattribute{\the\glslabeltok}{regular}{false}%
6653   }%
6654 }%
6655 }%
6656 }%
6657 {%

```

As short-sc-long style:

```

6658 \GlsXtrUseAbbrStyleFmts{short-sc-long}%
6659 }

```

short-sc

```

6660 \newabbreviationstyle{short-sc}%
6661 {%
6662 \renewcommand*\CustomAbbreviationFields{%
6663   name={\protect\glsabbrvscfont{\the\glsshorttok}},
6664   sort={\the\glsshorttok},
6665   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
6666   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
6667   text={\protect\glsabbrvscfont{\the\glsshorttok}},
6668   plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
6669   description={\the\glslongtok}}%
6670 \renewcommand*\GlsXtrPostNewAbbreviation{%
6671   \glsattribute{\the\glslabeltok}{regular}{true}}%
6672 }%
6673 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

6674 \renewcommand*\abbrvpluralsuffix{\protect\glxtrscsuffix}%
6675 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
6676 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
6677 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
6678 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

6679 \renewcommand*{\glxtrinlinefullformat}[2]{%
6680   \protect\glsfirstabbrvscfont{\glsaccessshort{##1}%
6681     \ifglxtrininsertinside##2\fi}%
6682   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6683   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
6684 }%
6685 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6686   \protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}%
6687     \ifglxtrininsertinside##2\fi}%
6688   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6689   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
6690 }%
6691 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6692   \protect\glsfirstabbrvscfont{\glsaccessshort{##1}%
6693     \ifglxtrininsertinside##2\fi}%
6694   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6695   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
6696 }%
6697 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6698   \protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}%
6699     \ifglxtrininsertinside##2\fi}%
6700   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6701   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
6702 }%

```

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.

```

6703 \renewcommand*{\glxtrfullformat}[2]{%
6704   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6705   \ifglxtrininsertinside\else##2\fi
6706 }%
6707 \renewcommand*{\glxtrfullplformat}[2]{%
6708   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6709   \ifglxtrininsertinside\else##2\fi
6710 }%
6711 \renewcommand*{\Glsxtrfullformat}[2]{%
6712   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6713   \ifglxtrininsertinside\else##2\fi
6714 }%
6715 \renewcommand*{\Glsxtrfullplformat}[2]{%
6716   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6717   \ifglxtrininsertinside\else##2\fi
6718 }%
6719 }

```

short-sc-nolong

```
6720 \letabbreviationstyle{short-sc-nolong}{short-sc}
```

short-sc-desc

```

6721 \newabbreviationstyle{short-sc-desc}%
6722 {%
6723   \renewcommand*{\CustomAbbreviationFields}{%
6724     name={\glxtrshortdescname},
6725     sort={\the\glsshorttok},
6726     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
6727     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
6728     text={\protect\glsabbrvscfont{\the\glsshorttok}},
6729     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
6730     description={\the\glslongtok}}%
6731 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6732   \glssetattribute{\the\glslabeltok}{regular}{true}}%
6733 }%
6734 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

6735 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
6736 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
6737 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
6738 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6739 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

6740 \renewcommand*{\glxtrinlinelinefullformat}[2]{%
6741   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6742   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6743   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
6744 }%
6745 \renewcommand*{\glxtrinlinelinefullplformat}[2]{%
6746   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6747   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6748   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
6749 }%
6750 \renewcommand*{\Glsxtrinlinelinefullformat}[2]{%
6751   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6752   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6753   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
6754 }%
6755 \renewcommand*{\Glsxtrinlinelinefullplformat}[2]{%
6756   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6757   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6758   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
6759 }%

```

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.

```

6760 \renewcommand*{\glxtrfullformat}[2]{%
6761   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6762   \ifglxtrininsertinside\else##2\fi
6763 }%
6764 \renewcommand*{\glxtrfullplformat}[2]{%

```

```

6765 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6766 \ifglxtrinsertinside\else##2\fi
6767 }%
6768 \renewcommand*{\Glsxtrfullformat}[2]{%
6769 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6770 \ifglxtrinsertinside\else##2\fi
6771 }%
6772 \renewcommand*{\Glsxtrfullplformat}[2]{%
6773 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6774 \ifglxtrinsertinside\else##2\fi
6775 }%
6776 }

```

-sc-nolong-desc

```
6777 \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 `\glsxtrshort`.

```

6778 \newabbreviationstyle{long-noshort-sc}%
6779 {%
6780 \renewcommand*{\CustomAbbreviationFields}{%
6781 name={\protect\glsabbrvscfont{\the\glsshorttok}},
6782 sort={\the\glsshorttok},
6783 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
6784 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
6785 text={\protect\glslongdefaultfont{\the\glslongtok}},
6786 plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
6787 description={\the\glslongtok}%
6788 }%
6789 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6790 \glssetattribute{\the\glslabeltok}{regular}{true}}%
6791 }%
6792 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

6793 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
6794 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
6795 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
6796 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6797 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

6798 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
6799 \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
6800 \ifglxtrinsertinside \else##2\fi
6801 }%
6802 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
6803 \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
6804 \ifglxtrinsertinside \else##2\fi
6805 }%

```

```

6806 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
6807   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
6808   \ifglsxtrininsertinside \else##2\fi
6809 }%
6810 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
6811   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
6812   \ifglsxtrininsertinside \else##2\fi
6813 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

6814 \renewcommand*{\glsxtrinlinefullformat}[2]{%
6815   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6816   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6817   \glsxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
6818 }%
6819 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
6820   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6821   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6822   \glsxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
6823 }%
6824 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6825   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6826   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6827   \glsxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
6828 }%
6829 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6830   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6831   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6832   \glsxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
6833 }%

```

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.

```

6834 \renewcommand*{\glsxtrfullformat}[2]{%
6835   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6836   \ifglsxtrininsertinside\else##2\fi
6837 }%
6838 \renewcommand*{\glsxtrfullplformat}[2]{%
6839   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6840   \ifglsxtrininsertinside\else##2\fi
6841 }%
6842 \renewcommand*{\Glsxtrfullformat}[2]{%
6843   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6844   \ifglsxtrininsertinside\else##2\fi
6845 }%
6846 \renewcommand*{\Glsxtrfullplformat}[2]{%
6847   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6848   \ifglsxtrininsertinside\else##2\fi
6849 }%
6850 }

```

long-sc Backward compatibility:

```
6851 \@glsxtr@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`.

```
6852 \newabbreviationstyle{long-noshort-sc-desc}%
```

```
6853 {%
```

```
6854 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
```

```
6855 }%
```

```
6856 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
6857 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
```

```
6858 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
```

```
6859 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
```

```
6860 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
```

```
6861 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
6862 \renewcommand*\glsxtrsubsequentfmt}[2]{%
```

```
6863 \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
```

```
6864 \ifglsxtrininsertinside \else##2\fi
```

```
6865 }%
```

```
6866 \renewcommand*\glsxtrsubsequentplfmt}[2]{%
```

```
6867 \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
```

```
6868 \ifglsxtrininsertinside \else##2\fi
```

```
6869 }%
```

```
6870 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
```

```
6871 \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
```

```
6872 \ifglsxtrininsertinside \else##2\fi
```

```
6873 }%
```

```
6874 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
```

```
6875 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
```

```
6876 \ifglsxtrininsertinside \else##2\fi
```

```
6877 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
6878 \renewcommand*\glsxtrininlinefullformat}[2]{%
```

```
6879 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
```

```
6880 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
```

```
6881 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
```

```
6882 }%
```

```
6883 \renewcommand*\glsxtrininlinefullplformat}[2]{%
```

```
6884 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
```

```
6885 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
```

```
6886 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
```

```
6887 }%
```

```
6888 \renewcommand*\Glsxtrininlinefullformat}[2]{%
```

```
6889 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
```

```
6890 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
```

```

6891 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
6892 }%
6893 \renewcommand*{\Glsxtrinlinelinefullplformat}[2]{%
6894 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6895 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6896 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
6897 }%

```

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.

```

6898 \renewcommand*{\glsxtrfullformat}[2]{%
6899 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6900 \ifglsxtrininsertinside\else##2\fi
6901 }%
6902 \renewcommand*{\glsxtrfullplformat}[2]{%
6903 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6904 \ifglsxtrininsertinside\else##2\fi
6905 }%
6906 \renewcommand*{\Glsxtrfullformat}[2]{%
6907 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6908 \ifglsxtrininsertinside\else##2\fi
6909 }%
6910 \renewcommand*{\Glsxtrfullplformat}[2]{%
6911 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6912 \ifglsxtrininsertinside\else##2\fi
6913 }%
6914 }

```

long-desc-sc Backward compatibility:

```
6915 \@glsxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}
```

ort-sc-footnote

```

6916 \newabbreviationstyle{short-sc-footnote}%
6917 {%
6918 \renewcommand*{\CustomAbbreviationFields}{%
6919 name={\protect\glsabbrvscfont{\the\glsshorttok}},
6920 sort={\the\glsshorttok},
6921 description={\the\glslongtok},%
6922 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%
6923 \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6924 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
6925 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}}%
6926 \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6927 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
6928 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.

```

6929 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6930 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%

```



```

6931 \glshasattribute{\the\glslabelltok}{regular}%
6932 {%
6933 \glissetattribute{\the\glslabelltok}{regular}{false}%
6934 }%
6935 {}%
6936 }%
6937 }%
6938 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

6939 \renewcommand*\abbrvpluralsuffix{\protect\glxtrscsuffix}%
6940 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
6941 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
6942 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
6943 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

6944 \renewcommand*\glsxtrfullformat}[2]{%
6945 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6946 \ifglsxtrininsertinside\else##2\fi
6947 \protect\glsxtrabbrvfootnote{##1}%
6948 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6949 }%
6950 \renewcommand*\glsxtrfullplformat}[2]{%
6951 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6952 \ifglsxtrininsertinside\else##2\fi
6953 \protect\glsxtrabbrvfootnote{##1}%
6954 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6955 }%
6956 \renewcommand*\Glsxtrfullformat}[2]{%
6957 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6958 \ifglsxtrininsertinside\else##2\fi
6959 \protect\glsxtrabbrvfootnote{##1}%
6960 {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
6961 }%
6962 \renewcommand*\Glsxtrfullplformat}[2]{%
6963 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6964 \ifglsxtrininsertinside\else##2\fi
6965 \protect\glsxtrabbrvfootnote{##1}%
6966 {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6967 }%

```

The first use full form and the inline full form use the short (long) style.

```

6968 \renewcommand*\glsxtrininlinefullformat}[2]{%
6969 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6970 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6971 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6972 }%
6973 \renewcommand*\glsxtrininlinefullplformat}[2]{%
6974 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6975 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%

```

```

6976   \glstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6977 }%
6978 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6979   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
6980   \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
6981   \glstrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6982 }%
6983 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6984   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
6985   \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
6986   \glstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6987 }%
6988 }

```

footnote-sc Backward compatibility:

```
6989 \@glstr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}
```

sc-postfootnote

```

6990 \newabbreviationstyle{short-sc-postfootnote}%
6991 {%
6992   \renewcommand*{\CustomAbbreviationFields}{%
6993     name={\protect\glsabbrvscfont{\the\glsshorttok}},
6994     sort={\the\glsshorttok},
6995     description={\the\glslongtok},%
6996     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
6997     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
6998     plural={\protect\glsabbrvscfont{\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.

```

6999 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7000   \csdef{glstrpostlink\glscategorylabel}{%
7001     \glstrifwasfirstuse
7002     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

7003     \glstrdopostpunc{\protect\glstrabbrvfootnote{\glslabel}%
7004     {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
7005   }%
7006   {}%
7007 }%
7008 \glshasattribute{\the\glslabeltok}{regular}%
7009 {%
7010   \glissetattribute{\the\glslabeltok}{regular}{false}%
7011 }%
7012 {}%
7013 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
7014 \renewcommand*\glxtrsetupfulldefs}{%
7015   \let\glxtrifwasfirstuse\@secondoftwo
7016 }%
7017 }%
7018 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
7019 \renewcommand*\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7020 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
7021 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
7022 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
7023 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
7024 \renewcommand*\glxtrfullformat}[2]{%
7025   \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7026   \ifglxtrininsertinside\else##2\fi
7027 }%
7028 \renewcommand*\glxtrfullplformat}[2]{%
7029   \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7030   \ifglxtrininsertinside\else##2\fi
7031 }%
7032 \renewcommand*\Glsxtrfullformat}[2]{%
7033   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7034   \ifglxtrininsertinside\else##2\fi
7035 }%
7036 \renewcommand*\Glsxtrfullplformat}[2]{%
7037   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7038   \ifglxtrininsertinside\else##2\fi
7039 }%
```

The first use full form and the inline full form use the short (long) style.

```
7040 \renewcommand*\glxtrinlinefullformat}[2]{%
7041   \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7042   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7043   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
7044 }%
7045 \renewcommand*\glxtrinlinefullplformat}[2]{%
7046   \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7047   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7048   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
7049 }%
7050 \renewcommand*\Glsxtrinlinefullformat}[2]{%
7051   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7052   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7053   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7054 }%
7055 \renewcommand*\Glsxtrinlinefullplformat}[2]{%

```

```

7056 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7057 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7058 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7059 }%
7060 }

```

postfootnote-sc Backward compatibility:

```
7061 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}
```

1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the relsize package, which must be loaded by the user. These styles all use:

`\glxtrsmfont` Maintained for backward compatibility.

```
7062 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}
```

`\glsabbrvsmfont` Added for consistent naming.

```
7063 \newcommand*{\glsabbrvsmfont}{\glxtrsmfont}
```

`sxtrfirstsmfont` Maintained for backward compatibility.

```
7064 \newcommand*{\sxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}
```

`irstabbrvsmfont` Added for consistent naming.

```
7065 \newcommand*{\irstabbrvsmfont}{\sxtrfirstsmfont}
```

and for the default short form suffix:

`\glxtrsmsuffix`

```
7066 \newcommand*{\glxtrsmsuffix}{\glxtrabbrvpluralsuffix}
```

`long-short-sm`

```
7067 \newabbreviationstyle{long-short-sm}%
```

```
7068 {%
```

```
7069 \renewcommand*{\CustomAbbreviationFields}{%
```

```
7070 name={\protect\glsabbrvsmfont{\the\glsshorttok}},
```

```
7071 sort={\the\glsshorttok},
```

```
7072 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
```

```
7073 \protect\glxtrfullsep{\the\glslabeltok}%
```

```
7074 \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
```

```
7075 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
```

```
7076 \protect\glxtrfullsep{\the\glslabeltok}%
```

```
7077 \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
```

```
7078 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}},%
```

```
7079 description={\the\glslongtok}}%
```

```
7080 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
7081 \glshasattribute{\the\glslabeltok}{regular}}%
```

```
7082 }%
```

```

7083     \glssetattribute{\the\glslabeltok}{regular}{false}%
7084   }%
7085   {}%
7086 }%
7087 }%
7088 {%
7089 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7090 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7091 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%

```

Use the default long fonts.

```

7092 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7093 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7094 \renewcommand*\glsxtrfullformat[2]{%
7095   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7096   \ifglsxtrinsertinside\else##2\fi
7097   \glsxtrfullsep{##1}%
7098   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7099 }%
7100 \renewcommand*\glsxtrfullplformat[2]{%
7101   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7102   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7103   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7104 }%
7105 \renewcommand*\Glsxtrfullformat[2]{%
7106   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7107   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7108   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7109 }%
7110 \renewcommand*\Glsxtrfullplformat[2]{%
7111   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7112   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7113   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7114 }%
7115 }

```

g-short-sm-desc

```

7116 \newabbreviationstyle{long-short-sm-desc}%
7117 {%
7118 \renewcommand*\CustomAbbreviationFields{%
7119   name={\glsxtrlongshortdescname},
7120   sort={\glsxtrlongshortdescsort},%
7121   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7122     \protect\glsxtrfullsep{\the\glslabeltok}%
7123     \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7124   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7125     \protect\glsxtrfullsep{\the\glslabeltok}%
7126     \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%

```

```

7127   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
7128   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
7129 }%

```

Unset the regular attribute if it has been set.

```

7130 \renewcommand*\GlsXtrPostNewAbbreviation{%
7131   \glsattribute{\the\glslabeltok}{regular}%
7132   {%
7133     \glssetattribute{\the\glslabeltok}{regular}{false}%
7134   }%
7135 }%
7136 }%
7137 }%
7138 {%

```

As long-short-sm style:

```

7139 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
7140 }

```

short-sm-long Now the short (long) version

```

7141 \newabbreviationstyle{short-sm-long}%
7142 {%
7143   \renewcommand*\CustomAbbreviationFields{%
7144     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7145     sort={\the\glsshorttok},
7146     description={\the\glslongtok},%
7147     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}}%
7148     \protect\glsxtrfullsep{\the\glslabeltok}%
7149     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7150     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}%
7151     \protect\glsxtrfullsep{\the\glslabeltok}%
7152     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7153     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7154 \renewcommand*\GlsXtrPostNewAbbreviation{%
7155   \glsattribute{\the\glslabeltok}{regular}%
7156   {%
7157     \glssetattribute{\the\glslabeltok}{regular}{false}%
7158   }%
7159 }%
7160 }%
7161 }%
7162 {%
7163   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7164   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7165   \renewcommand*\abbrvpluralsuffix{\protect\glsxtrrmsuffix}%
7166   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7167   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7168 \renewcommand*{\glxtrfullformat}[2]{%
7169   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7170   \ifglxtrininsertinside\else##2\fi
7171   \glxtrfullsep{##1}%
7172   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7173 }%
7174 \renewcommand*{\glxtrfullplformat}[2]{%
7175   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7176   \ifglxtrininsertinside\else##2\fi
7177   \glxtrfullsep{##1}%
7178   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7179 }%
7180 \renewcommand*{\Glsxtrfullformat}[2]{%
7181   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7182   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7183   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7184 }%
7185 \renewcommand*{\Glsxtrfullplformat}[2]{%
7186   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7187   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7188   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7189 }%
7190 }

```

rt-sm-long-desc As before but user provides description

```

7191 \newabbreviationstyle{short-sm-long-desc}%
7192 {%
7193   \renewcommand*{\CustomAbbreviationFields}{%
7194     name={\glxtrshortlongdescname},
7195     sort={\glxtrshortlongdescsort},
7196     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}}%
7197     \protect\glxtrfullsep{\the\glslabeltok}}%
7198     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7199     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}%
7200     \protect\glxtrfullsep{\the\glslabeltok}}%
7201     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7202     text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
7203     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
7204 }%

```

Unset the regular attribute if it has been set.

```

7205 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7206   \glsattribute{\the\glslabeltok}{regular}%
7207   {%
7208     \glssetattribute{\the\glslabeltok}{regular}{false}%
7209   }%
7210 }%
7211 }%
7212 }%
7213 {%

```

As short-sm-long style:

```
7214 \GlsXtrUseAbbrStyleFmts{short-sm-long}%  
7215 }
```

short-sm

```
7216 \newabbreviationstyle{short-sm}%  
7217 {%  
7218 \renewcommand*{\CustomAbbreviationFields}{%  
7219   name={\protect\glsabbrvsmfont{\the\glsshorttok}},  
7220   sort={\the\glsshorttok},  
7221   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},  
7222   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},  
7223   text={\protect\glsabbrvsmfont{\the\glsshorttok}},  
7224   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},  
7225   description={\the\glslongtok}}%  
7226 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
7227   \glssetattribute{\the\glslabeltok}{regular}{true}}%  
7228 }%  
7229 {%  
7230 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%  
7231 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%  
7232 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%  
7233 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%  
7234 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
7235 \renewcommand*\glsxtrinlinefullformat}[2]{%  
7236   \protect\glsfirstabbrvsmfont{\glsaccesssshort{##1}}%  
7237   \ifglsxtrininsertinside##2\fi}%  
7238   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%  
7239   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%  
7240 }%  
7241 \renewcommand*\glsxtrinlinefullplformat}[2]{%  
7242   \protect\glsfirstabbrvsmfont{\glsaccesssshortpl{##1}}%  
7243   \ifglsxtrininsertinside##2\fi}%  
7244   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%  
7245   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%  
7246 }%  
7247 \renewcommand*\Glsxtrinlinefullformat}[2]{%  
7248   \protect\glsfirstabbrvsmfont{\glsaccesssshort{##1}}%  
7249   \ifglsxtrininsertinside##2\fi}%  
7250   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%  
7251   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%  
7252 }%  
7253 \renewcommand*\Glsxtrinlinefullplformat}[2]{%  
7254   \protect\glsfirstabbrvsmfont{\glsaccesssshortpl{##1}}%  
7255   \ifglsxtrininsertinside##2\fi}%  
7256   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%  
7257   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%  
7258 }%
```


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.

```

7259 \renewcommand*\glxtrfullformat}[2]{%
7260   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7261   \ifglxtrinsertinside\else##2\fi
7262 }%
7263 \renewcommand*\glxtrfullplformat}[2]{%
7264   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7265   \ifglxtrinsertinside\else##2\fi
7266 }%
7267 \renewcommand*\Glsxtrfullformat}[2]{%
7268   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7269   \ifglxtrinsertinside\else##2\fi
7270 }%
7271 \renewcommand*\Glsxtrfullplformat}[2]{%
7272   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7273   \ifglxtrinsertinside\else##2\fi
7274 }%
7275 }

```

short-sm-nolong

```

7276 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

7277 \newabbreviationstyle{short-sm-desc}{%
7278 {%
7279   \renewcommand*\CustomAbbreviationFields}{%
7280     name={\glxtrshortdesname},
7281     sort={\the\glsshorttok},
7282     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
7283     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
7284     text={\protect\glsabbrvsmfont{\the\glsshorttok}},
7285     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
7286     description={\the\glslongtok}}%
7287 \renewcommand*\GlsXtrPostNewAbbreviation}{%
7288   \glssetattribute{\the\glslabeltok}{regular}{true}}%
7289 }%
7290 {%
7291   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7292   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7293   \renewcommand*\abbrvpluralsuffix{\protect\glxtrrmsuffix}%
7294   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7295   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7296 \renewcommand*\glxtrinlinefullformat}[2]{%
7297   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7298   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7299   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%

```

```

7300 }%
7301 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7302   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7303   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7304   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7305 }%
7306 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7307   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7308   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7309   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7310 }%
7311 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7312   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7313   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7314   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7315 }%

```

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.

```

7316 \renewcommand*{\glxtrfullformat}[2]{%
7317   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7318   \ifglxtrininsertinside\else##2\fi
7319 }%
7320 \renewcommand*{\glxtrfullplformat}[2]{%
7321   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7322   \ifglxtrininsertinside\else##2\fi
7323 }%
7324 \renewcommand*{\Glsxtrfullformat}[2]{%
7325   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7326   \ifglxtrininsertinside\else##2\fi
7327 }%
7328 \renewcommand*{\Glsxtrfullplformat}[2]{%
7329   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7330   \ifglxtrininsertinside\else##2\fi
7331 }%
7332 }

```

-sm-nolong-desc

```
7333 \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`.

```

7334 \newabbreviationstyle{long-noshort-sm}%
7335 {%
7336   \renewcommand*{\CustomAbbreviationFields}{%
7337     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7338     sort={\the\glsshorttok},
7339     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7340     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},

```

```

7341     text={\protect\glslongdefaultfont{\the\glslongtok}},
7342     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7343     description={\the\glslongtok}%
7344 }%
7345 \renewcommand*\GlsXtrPostNewAbbreviation{%
7346   \glssetattribute{\the\glslabeltok}{regular}{true}}%
7347 }%
7348 {%
7349 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7350 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7351 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%
7352 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7353 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7354 \renewcommand*\glsxtrsubsequentfmt[2]{%
7355   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7356   \ifglsxtrininsertinside \else##2\fi
7357 }%
7358 \renewcommand*\glsxtrsubsequentplfmt[2]{%
7359   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7360   \ifglsxtrininsertinside \else##2\fi
7361 }%
7362 \renewcommand*\Glsxtrsubsequentfmt[2]{%
7363   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7364   \ifglsxtrininsertinside \else##2\fi
7365 }%
7366 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
7367   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7368   \ifglsxtrininsertinside \else##2\fi
7369 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7370 \renewcommand*\glsxtrinlinefullformat[2]{%
7371   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7372   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7373   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7374 }%
7375 \renewcommand*\glsxtrinlinefullplformat[2]{%
7376   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7377   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7378   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7379 }%
7380 \renewcommand*\Glsxtrinlinefullformat[2]{%
7381   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7382   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7383   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7384 }%
7385 \renewcommand*\Glsxtrinlinefullplformat[2]{%
7386   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%

```

```

7387     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7388     \glxtrparen{\protect\glsfirstabbrvsmfont{\glssaccessshortpl{##1}}}%
7389 }%

```

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.

```

7390 \renewcommand*\glxtrfullformat}[2]{%
7391   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7392   \ifglxtrinsertinside\else##2\fi
7393 }%
7394 \renewcommand*\glxtrfullplformat}[2]{%
7395   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7396   \ifglxtrinsertinside\else##2\fi
7397 }%
7398 \renewcommand*\Glsxtrfullformat}[2]{%
7399   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7400   \ifglxtrinsertinside\else##2\fi
7401 }%
7402 \renewcommand*\Glsxtrfullplformat}[2]{%
7403   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7404   \ifglxtrinsertinside\else##2\fi
7405 }%
7406 }

```

long-sm Backward compatibility:

```

7407 \@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 `\glssshort`.

```

7408 \newabbreviationstyle{long-noshort-sm-desc}%
7409 {%
7410   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7411 }%
7412 {%
7413   \renewcommand*\glssabbrvfont[1]{\glssabbrvsmfont{##1}}%
7414   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7415   \renewcommand*\abbrvpluralsuffix{\protect\glsxtrrmsuffix}%
7416   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7417   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7418 \renewcommand*\glxtrsubsequentfmt}[2]{%
7419   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7420   \ifglxtrinsertinside \else##2\fi
7421 }%
7422 \renewcommand*\glxtrsubsequentplfmt}[2]{%
7423   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7424   \ifglxtrinsertinside \else##2\fi
7425 }%
7426 \renewcommand*\Glsxtrsubsequentfmt}[2]{%

```

```

7427 \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside ##2\fi}%
7428 \ifglsxtrinsertinside \else##2\fi
7429 }%
7430 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7431 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside ##2\fi}%
7432 \ifglsxtrinsertinside \else##2\fi
7433 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7434 \renewcommand*{\glsxtrinlinelinefullformat}[2]{%
7435 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7436 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7437 \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7438 }%
7439 \renewcommand*{\glsxtrinlinelinefullplformat}[2]{%
7440 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7441 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7442 \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7443 }%
7444 \renewcommand*{\Glsxtrinlinelinefullformat}[2]{%
7445 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7446 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7447 \glsxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
7448 }%
7449 \renewcommand*{\Glsxtrinlinelinefullplformat}[2]{%
7450 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7451 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7452 \glsxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
7453 }%

```

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.

```

7454 \renewcommand*{\glsxtrfullformat}[2]{%
7455 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7456 \ifglsxtrinsertinside\else##2\fi
7457 }%
7458 \renewcommand*{\glsxtrfullplformat}[2]{%
7459 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7460 \ifglsxtrinsertinside\else##2\fi
7461 }%
7462 \renewcommand*{\Glsxtrfullformat}[2]{%
7463 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7464 \ifglsxtrinsertinside\else##2\fi
7465 }%
7466 \renewcommand*{\Glsxtrfullplformat}[2]{%
7467 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7468 \ifglsxtrinsertinside\else##2\fi
7469 }%
7470 }

```

long-desc-sm Backward compatibility:

```
7471 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}
```

ort-sm-footnote

```
7472 \newabbreviationstyle{short-sm-footnote}%
7473 {%
7474   \renewcommand*{\CustomAbbreviationFields}{%
7475     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7476     sort={\the\glsshorttok},
7477     description={\the\glslongtok},%
7478     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
7479       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7480       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
7481     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
7482       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7483       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
7484     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.

```
7485   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7486     \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
7487     \glsattribute{\the\glslabeltok}{regular}%
7488     {%
7489       \glssetattribute{\the\glslabeltok}{regular}{false}%
7490     }%
7491   }%
7492 }%
7493 }%
7494 {%
7495   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
7496   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
7497   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrmsuffix}%
7498   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7499   \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
7500   \renewcommand*{\glxtrfullformat}[2]{%
7501     \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7502     \ifglxtrinsertinside\else##2\fi
7503     \protect\glxtrabbrvfootnote{##1}%
7504     {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7505   }%
7506   \renewcommand*{\glxtrfullplformat}[2]{%
7507     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7508     \ifglxtrinsertinside\else##2\fi
7509     \protect\glxtrabbrvfootnote{##1}%
7510     {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7511   }%
7512   \renewcommand*{\GlsXtrfullformat}[2]{%
```

```

7513 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7514 \ifglxtrinsertinside\else##2\fi
7515 \protect\glxtrabbrvfootnote{##1}%
7516 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7517 }%
7518 \renewcommand*{\Glsxtrfullplformat}[2]{%
7519 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7520 \ifglxtrinsertinside\else##2\fi
7521 \protect\glxtrabbrvfootnote{##1}%
7522 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7523 }%

```

The first use full form and the inline full form use the short (long) style.

```

7524 \renewcommand*{\glsxtrinlinelinefullformat}[2]{%
7525 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7526 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7527 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7528 }%
7529 \renewcommand*{\glsxtrinlinelinefullplformat}[2]{%
7530 \glsfirstabbrvsmfont{\glsaccessshorttpl{##1}\ifglxtrinsertinside##2\fi}%
7531 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7532 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7533 }%
7534 \renewcommand*{\Glsxtrinlinelinefullformat}[2]{%
7535 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7536 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7537 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7538 }%
7539 \renewcommand*{\Glsxtrinlinelinefullplformat}[2]{%
7540 \glsfirstabbrvsmfont{\Glsaccessshorttpl{##1}\ifglxtrinsertinside##2\fi}%
7541 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7542 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7543 }%
7544 }

```

footnote-sm Backward compatibility:

```
7545 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}
```

sm-postfootnote

```

7546 \newabbreviationstyle{short-sm-postfootnote}%
7547 {%
7548 \renewcommand*{\CustomAbbreviationFields}{%
7549 name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7550 sort={\the\glsshorttok},
7551 description={\the\glslongtok},%
7552 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
7553 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
7554 plural={\protect\glsabbrvsmfont{\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.

```
7555 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7556   \csdef{glxtrpostlink\glscategorylabel}{%
7557     \glxtrifwasfirstuse
7558     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
7559     \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
7560     {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
7561   }%
7562   {}%
7563 }%
7564 \glshasattribute{\the\glslabelltok}{regular}%
7565 {%
7566   \glissetattribute{\the\glslabelltok}{regular}{false}%
7567   }%
7568   {}%
7569 }%
```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
7570 \renewcommand*{\glxtrsetupfulldefs}{%
7571   \let\glxtrifwasfirstuse\@secondoftwo
7572 }%
7573 }%
7574 {%
7575 \renewcommand*{\glsabbrvfont[1]{\glsabbrvsmfont{##1}}}%
7576 \renewcommand*{\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}}%
7577 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrrmsuffix}%
7578 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7579 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
7580 \renewcommand*{\glxtrfullformat}[2]{%
7581   \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7582   \ifglxtrininsertinside\else##2\fi
7583 }%
7584 \renewcommand*{\glxtrfullplformat}[2]{%
7585   \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7586   \ifglxtrininsertinside\else##2\fi
7587 }%
7588 \renewcommand*{\Glsxtrfullformat}[2]{%
7589   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7590   \ifglxtrininsertinside\else##2\fi
7591 }%
7592 \renewcommand*{\Glsxtrfullplformat}[2]{%
7593   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7594   \ifglxtrininsertinside\else##2\fi
```


7595 }%

The first use full form and the inline full form use the short (long) style.

```
7596 \renewcommand*\glsxtrinlinefullformat}[2]{%
7597   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7598   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7599   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7600 }%
7601 \renewcommand*\glsxtrinlinefullplformat}[2]{%
7602   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7603   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7604   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7605 }%
7606 \renewcommand*\Glsxtrinlinefullformat}[2]{%
7607   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7608   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7609   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7610 }%
7611 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
7612   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7613   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7614   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7615 }%
7616 }
```

postfootnote-sm Backward compatibility:

```
7617 \@glsxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}
```

1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```
7618 \newcommand*\glsabbrvemfont}[1]{\emph{##1}}%
```

`irstabbrvemfont`

```
7619 \newcommand*\glsfirstabbrvemfont}[1]{\glsabbrvemfont{##1}}%
```

The default short form suffix:

`\glsxtremsuffix`

```
7620 \newcommand*\glsxtremsuffix{\glsxtrabbrvpluralsuffix}
```

`firstlongemfont` Only used by the “long-em” styles.

```
7621 \newcommand*\glsfirstlongemfont}[1]{\glslongemfont{##1}}%
```

`\glslongemfont` Only used by the “long-em” styles.

```
7622 \newcommand*\glslongemfont}[1]{\emph{##1}}%
```

long-short-em The long form is just set in the default long font.

```
7623 \newabbreviationstyle{long-short-em}%
7624 {%
7625   \renewcommand*{\CustomAbbreviationFields}{%
7626     name={\protect\glsabbrvemfont{\the\glsshorttok}},
7627     sort={\the\glsshorttok},
7628     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7629       \protect\glsxtrfullsep{\the\glslabeltok}%
7630       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
7631     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7632       \protect\glsxtrfullsep{\the\glslabeltok}%
7633       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
7634     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
7635     description={\the\glslongtok}}%
7636 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7637   \glsattribute{\the\glslabeltok}{regular}%
7638   {%
7639     \glssetattribute{\the\glslabeltok}{regular}{false}%
7640   }%
7641   {}}%
7642 }%
7643 }%
7644 {%
7645 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
7646 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
7647 \renewcommand*\abbrvpluralsuffix{\protect\glsxtremsuffix}%
```

Use the default long fonts.

```
7648 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7649 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
7650 \renewcommand*{\glsxtrfullformat}[2]{%
7651   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinertinside##2\fi}%
7652   \ifglsxtrinertinside\else##2\fi
7653   \glsxtrfullsep{##1}%
7654   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
7655 }%
7656 \renewcommand*{\glsxtrfullplformat}[2]{%
7657   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinertinside##2\fi}%
7658   \ifglsxtrinertinside\else##2\fi\glsxtrfullsep{##1}%
7659   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
7660 }%
7661 \renewcommand*{\Glsxtrfullformat}[2]{%
7662   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinertinside##2\fi}%
7663   \ifglsxtrinertinside\else##2\fi\glsxtrfullsep{##1}%
7664   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
7665 }%
7666 \renewcommand*{\Glsxtrfullplformat}[2]{%
7667   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinertinside##2\fi}%
```

```

7668   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7669   \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
7670 }%
7671 }

```

g-short-em-desc

```

7672 \newabbreviationstyle{long-short-em-desc}%
7673 {%
7674   \renewcommand*{\CustomAbbreviationFields}{%
7675     name={\glxtrlongshortdescname},
7676     sort={\glxtrlongshortdescsort},%
7677     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7678       \protect\glxtrfullsep{\the\glslabeltok}%
7679       \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
7680     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7681       \protect\glxtrfullsep{\the\glslabeltok}%
7682       \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
7683     text={\protect\glssabbrvemfont{\the\glsshorttok}},%
7684     plural={\protect\glssabbrvemfont{\the\glsshortpltok}}%
7685   }%

```

Unset the regular attribute if it has been set.

```

7686   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7687     \glshasattribute{\the\glslabeltok}{regular}%
7688     {%
7689       \glissetattribute{\the\glslabeltok}{regular}{false}%
7690     }%
7691   }%
7692 }%
7693 }%
7694 {%

```

As long-short-em style:

```

7695   \GlsXtrUseAbbrStyleFmts{long-short-em}%
7696 }

```

ong-em-short-em

```

7697 \newabbreviationstyle{long-em-short-em}%
7698 {%
7699   \glslongemfont is used in the description since \glstdesc doesn't set the style.
7699   \renewcommand*{\CustomAbbreviationFields}{%
7700     name={\protect\glssabbrvemfont{\the\glsshorttok}},
7701     sort={\the\glsshorttok},
7702     first={\protect\glsfirstlongemfont{\the\glslongtok}%
7703       \protect\glxtrfullsep{\the\glslabeltok}%
7704       \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
7705     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
7706       \protect\glxtrfullsep{\the\glslabeltok}%
7707       \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%

```

```

7708 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
7709 description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

7710 \renewcommand*\GlsXtrPostNewAbbreviation}{%
7711   \glsattribute{\the\glslabeltok}{regular}%
7712   {%
7713     \glsattribute{\the\glslabeltok}{regular}{false}%
7714   }%
7715   {}%
7716 }%
7717 }%
7718 {%
7719 \renewcommand*\abbrvpluralsuffix{\protect\glsxtremsuffix}%
7720 \renewcommand*\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
7721 \renewcommand*\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
7722 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
7723 \renewcommand*\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7724 \renewcommand*\glsxtrfullformat}[2]{%
7725   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7726   \ifglsxtrininsertinside\else##2\fi
7727   \glsxtrfullsep{##1}%
7728   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
7729 }%
7730 \renewcommand*\glsxtrfullplformat}[2]{%
7731   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7732   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7733   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
7734 }%
7735 \renewcommand*\Glsxtrfullformat}[2]{%
7736   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7737   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7738   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
7739 }%
7740 \renewcommand*\Glsxtrfullplformat}[2]{%
7741   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7742   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7743   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
7744 }%
7745 }

```

m-short-em-desc

```

7746 \newabbreviationstyle{long-em-short-em-desc}%
7747 {%
7748 \renewcommand*\CustomAbbreviationFields}{%
7749   name={\glsxtrlongshortdescname},
7750   sort={\glsxtrlongshortdescsort},%
7751   first={\protect\glsfirstlongemfont{\the\glslongtok}}%

```

```

7752 \protect\glxtrfullsep{\the\glslabeltok}%
7753 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
7754 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
7755 \protect\glxtrfullsep{\the\glslabeltok}%
7756 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
7757 text={\protect\glsabbrvemfont{\the\glsshorttok}}},%
7758 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
7759 }%

```

Unset the regular attribute if it has been set.

```

7760 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7761 \glshasattribute{\the\glslabeltok}{regular}%
7762 {%
7763 \glissetattribute{\the\glslabeltok}{regular}{false}%
7764 }%
7765 {}%
7766 }%
7767 }%
7768 {%
7769 \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
7770 }

```

short-em-long Now the short (long) version

```

7771 \newabbreviationstyle{short-em-long}%
7772 {%
7773 \renewcommand*{\CustomAbbreviationFields}{%
7774 name={\protect\glsabbrvemfont{\the\glsshorttok}},
7775 sort={\the\glsshorttok},
7776 description={\the\glslongtok},%
7777 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
7778 \protect\glxtrfullsep{\the\glslabeltok}%
7779 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7780 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
7781 \protect\glxtrfullsep{\the\glslabeltok}%
7782 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7783 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7784 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7785 \glshasattribute{\the\glslabeltok}{regular}%
7786 {%
7787 \glissetattribute{\the\glslabeltok}{regular}{false}%
7788 }%
7789 {}%
7790 }%
7791 }%
7792 {%

```

Mostly as short-long style:

```

7793 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%

```

```

7794 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
7795 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
7796 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7797 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7798 \renewcommand*\glsxtrfullformat}[2]{%
7799   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7800   \ifglsxtrininsertinside\else##2\fi
7801   \glsxtrfullsep{##1}%
7802   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7803 }%
7804 \renewcommand*\glsxtrfullplformat}[2]{%
7805   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7806   \ifglsxtrininsertinside\else##2\fi
7807   \glsxtrfullsep{##1}%
7808   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7809 }%
7810 \renewcommand*\Glsxtrfullformat}[2]{%
7811   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7812   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7813   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7814 }%
7815 \renewcommand*\Glsxtrfullplformat}[2]{%
7816   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7817   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7818   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7819 }%
7820 }

```

rt-em-long-desc As before but user provides description

```

7821 \newabbreviationstyle{short-em-long-desc}%
7822 {%
7823   \renewcommand*\CustomAbbreviationFields{%
7824     name={\glsxtrshortlongdescname},
7825     sort={\glsxtrshortlongdescsort},
7826     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
7827       \protect\glsxtrfullsep{\the\glslabeltok}%
7828       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7829     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
7830       \protect\glsxtrfullsep{\the\glslabeltok}%
7831       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7832     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
7833     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
7834   }%

```

Unset the regular attribute if it has been set.

```

7835 \renewcommand*\GlsXtrPostNewAbbreviation{%
7836   \glsattribute{\the\glslabeltok}{regular}%
7837   {%

```

```

7838     \glsetattribute{\the\glslabeltok}{regular}{false}%
7839   }%
7840   {}%
7841 }%
7842 }%
7843 {%
7844   \GlsXtrUseAbbrStyleFmts{short-em-long}%
7845 }

```

hort-em-long-em

```

7846 \newabbreviationstyle{short-em-long-em}%
7847 {%

```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```

7848 \renewcommand*{\CustomAbbreviationFields}{%
7849   name={\protect\glsabbrvemfont{\the\glsshorttok}},
7850   sort={\the\glsshorttok},
7851   description={\protect\glslongemfont{\the\glslongtok}},%
7852   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
7853     \protect\glsxtrfullsep{\the\glslabeltok}}%
7854   \glxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
7855   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
7856     \protect\glsxtrfullsep{\the\glslabeltok}}%
7857   \glxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
7858   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7859 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7860   \glshasattribute{\the\glslabeltok}{regular}%
7861   {%
7862     \glsetattribute{\the\glslabeltok}{regular}{false}%
7863     }%
7864     {}%
7865   }%
7866 }%
7867 {%
7868 \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
7869 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
7870 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
7871 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
7872 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7873 \renewcommand*{\glsxtrfullformat}[2]{%
7874   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7875   \ifglxtrininsertinside\else##2\fi
7876   \glsxtrfullsep{##1}}%
7877   \glxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
7878 }%
7879 \renewcommand*{\glsxtrfullplformat}[2]{%

```

```

7880 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7881 \ifglxtrinsertinside\else##2\fi
7882 \glxtrfullsep{##1}%
7883 \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
7884 }%
7885 \renewcommand*{\Glsxtrfullformat}[2]{%
7886 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7887 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7888 \glxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
7889 }%
7890 \renewcommand*{\Glsxtrfullplformat}[2]{%
7891 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7892 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7893 \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
7894 }%
7895 }

```

em-long-em-desc

```

7896 \newabbreviationstyle{short-em-long-em-desc}%
7897 {%
7898 \renewcommand*{\CustomAbbreviationFields}{%
7899 name={\glxtrshortlongdescname},%
7900 sort={\glxtrshortlongdescsort},%
7901 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
7902 \protect\glxtrfullsep{\the\glslabeltok}%
7903 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
7904 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
7905 \protect\glxtrfullsep{\the\glslabeltok}%
7906 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
7907 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
7908 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
7909 }%

```

Unset the regular attribute if it has been set.

```

7910 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7911 \glsattribute{\the\glslabeltok}{regular}%
7912 {%
7913 \glssetattribute{\the\glslabeltok}{regular}{false}%
7914 }%
7915 {}%
7916 }%
7917 }%
7918 {%
7919 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
7920 }

```

short-em

```

7921 \newabbreviationstyle{short-em}%
7922 {%
7923 \renewcommand*{\CustomAbbreviationFields}{%

```



```

7924   name={\protect\glsabbrvemfont{\the\glsshorttok}},
7925   sort={\the\glsshorttok},
7926   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
7927   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
7928   text={\protect\glsabbrvemfont{\the\glsshorttok}},
7929   plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
7930   description={\the\glslongtok}}%
7931 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7932   \glssetattribute{\the\glslabeltok}{regular}{true}}%
7933 }%
7934 {%
7935 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
7936 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
7937 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
7938 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7939 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7940 \renewcommand*{\glsxtrinlinefullformat}[2]{%
7941   \protect\glsfirstabbrvemfont{\glsaccessshort{##1}}%
7942   \ifglsxtrininsertinside##2\fi}%
7943 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7944 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7945 }%
7946 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7947   \protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}%
7948   \ifglsxtrininsertinside##2\fi}%
7949 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7950 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7951 }%
7952 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7953   \protect\glsfirstabbrvemfont{\glsaccessshort{##1}}%
7954   \ifglsxtrininsertinside##2\fi}%
7955 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7956 \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7957 }%
7958 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7959   \protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}%
7960   \ifglsxtrininsertinside##2\fi}%
7961 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7962 \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7963 }%

```

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.

```

7964 \renewcommand*{\glsxtrfullformat}[2]{%
7965   \glsfirstabbrvemfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7966   \ifglsxtrininsertinside\else##2\fi
7967 }%
7968 \renewcommand*{\glsxtrfullplformat}[2]{%

```

```

7969   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7970   \ifglxtrinsertinside\else##2\fi
7971 }%
7972 \renewcommand*{\Glsxtrfullformat}[2]{%
7973   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7974   \ifglxtrinsertinside\else##2\fi
7975 }%
7976 \renewcommand*{\Glsxtrfullplformat}[2]{%
7977   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7978   \ifglxtrinsertinside\else##2\fi
7979 }%
7980 }

```

short-em-nolong

```
7981 \letabbreviationstyle{short-em-nolong}{short-em}
```

short-em-desc

```

7982 \newabbreviationstyle{short-em-desc}%
7983 {%
7984   \renewcommand*{\CustomAbbreviationFields}{%
7985     name={\glsxtrshortdescname},
7986     sort={\the\glsshorttok},
7987     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
7988     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
7989     text={\protect\glsabbrvemfont{\the\glsshorttok}},
7990     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
7991     description={\the\glslongtok}}%
7992   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7993     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7994 }%
7995 {%
7996   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
7997   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
7998   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
7999   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8000   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8001   \renewcommand*{\glsxtrinlinelinefullformat}[2]{%
8002     \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8003     \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8004     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8005   }%
8006   \renewcommand*{\glsxtrinlinelinefullplformat}[2]{%
8007     \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8008     \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8009     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8010   }%
8011   \renewcommand*{\Glsxtrinlinelinefullformat}[2]{%
8012     \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%

```

```

8013 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8014 \glxtrparen{\glstfirstlongdefaultfont{\glstaccesslong{##1}}}%
8015 }%
8016 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8017 \glstfirstabbrvemfont{\glstaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8018 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8019 \glxtrparen{\glstfirstlongdefaultfont{\glstaccesslongpl{##1}}}%
8020 }%

```

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.

```

8021 \renewcommand*{\glxtrfullformat}[2]{%
8022 \glstfirstabbrvemfont{\glstaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8023 \ifglxtrinsertinside\else##2\fi
8024 }%
8025 \renewcommand*{\glxtrfullplformat}[2]{%
8026 \glstfirstabbrvemfont{\glstaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8027 \ifglxtrinsertinside\else##2\fi
8028 }%
8029 \renewcommand*{\Glsxtrfullformat}[2]{%
8030 \glstfirstabbrvemfont{\glstaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8031 \ifglxtrinsertinside\else##2\fi
8032 }%
8033 \renewcommand*{\Glsxtrfullplformat}[2]{%
8034 \glstfirstabbrvemfont{\glstaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8035 \ifglxtrinsertinside\else##2\fi
8036 }%
8037 }

```

-em-nolong-desc

```
8038 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}
```

long-noshort-em The short form is explicitly invoked through commands like `\glsshort`.

```

8039 \newabbreviationstyle{long-noshort-em}%
8040 {%
8041 \renewcommand*{\CustomAbbreviationFields}{%
8042 name={\protect\glstabbrvemfont{\the\glstshorttok}},
8043 sort={\the\glstshorttok},
8044 first={\protect\glstfirstlongdefaultfont{\the\glstlongtok}},
8045 firstplural={\protect\glstfirstlongdefaultfont{\the\glstlongpltok}},
8046 text={\protect\glstlongdefaultfont{\the\glstlongtok}},
8047 plural={\protect\glstlongdefaultfont{\the\glstlongpltok}},%
8048 description={\the\glstlongtok}%
8049 }%
8050 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8051 \glstsetattribute{\the\glstlabeltok}{regular}{true}}%
8052 }%
8053 {%
8054 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%

```

```

8055 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
8056 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
8057 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8058 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8059 \renewcommand*\glsxtrsubsequentfmt[2]{%
8060   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8061   \ifglsxtrininsertinside \else##2\fi
8062 }%
8063 \renewcommand*\glsxtrsubsequentplfmt[2]{%
8064   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8065   \ifglsxtrininsertinside \else##2\fi
8066 }%
8067 \renewcommand*\Glsxtrsubsequentfmt[2]{%
8068   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8069   \ifglsxtrininsertinside \else##2\fi
8070 }%
8071 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
8072   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8073   \ifglsxtrininsertinside \else##2\fi
8074 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8075 \renewcommand*\glsxtrinlinefullformat[2]{%
8076   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8077   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8078   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8079 }%
8080 \renewcommand*\glsxtrinlinefullplformat[2]{%
8081   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8082   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8083   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8084 }%
8085 \renewcommand*\Glsxtrinlinefullformat[2]{%
8086   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8087   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8088   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8089 }%
8090 \renewcommand*\Glsxtrinlinefullplformat[2]{%
8091   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8092   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8093   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8094 }%

```

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.

```

8095 \renewcommand*\glsxtrfullformat[2]{%
8096   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8097   \ifglsxtrininsertinside\else##2\fi
8098 }%

```

```

8099 \renewcommand*\glxtrfullplformat}[2]{%
8100   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8101   \ifglxtrinsertinside\else##2\fi
8102 }%
8103 \renewcommand*\Glsxtrfullformat}[2]{%
8104   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8105   \ifglxtrinsertinside\else##2\fi
8106 }%
8107 \renewcommand*\Glsxtrfullplformat}[2]{%
8108   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8109   \ifglxtrinsertinside\else##2\fi
8110 }%
8111 }

```

long-em Backward compatibility:

```
8112 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}
```

g-em-noshort-em The short form is explicitly invoked through commands like `\glsshort`.

```

8113 \newabbreviationstyle{long-em-noshort-em}%
8114 {%
8115   \renewcommand*\CustomAbbreviationFields{%
8116     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8117     sort={\the\glsshorttok},
8118     first={\protect\glsfirstlongemfont{\the\glslongtok}},
8119     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
8120     text={\protect\glslongemfont{\the\glslongtok}},
8121     plural={\protect\glslongemfont{\the\glslongpltok}},%
8122     description={\protect\glslongemfont{\the\glslongtok}}%
8123   }%
8124   \renewcommand*\GlsXtrPostNewAbbreviation{%
8125     \glssetattribute{\the\glslabeltok}{regular}{true}}%
8126 }%
8127 {%
8128   \renewcommand*\abbrvpluralsuffix{\protect\glsxtremsuffix}%
8129   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
8130   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
8131   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
8132   \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8133 \renewcommand*\glxtrsubsequentfmt}[2]{%
8134   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8135   \ifglxtrinsertinside \else##2\fi
8136 }%
8137 \renewcommand*\glxtrsubsequentplfmt}[2]{%
8138   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8139   \ifglxtrinsertinside \else##2\fi
8140 }%
8141 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
8142   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%

```

```

8143 \ifglxtrinsertinside \else##2\fi
8144 }%
8145 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
8146 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8147 \ifglxtrinsertinside \else##2\fi
8148 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8149 \renewcommand*\glxtrinlinefullformat}[2]{%
8150 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8151 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8152 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8153 }%
8154 \renewcommand*\glxtrinlinefullplformat}[2]{%
8155 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8156 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8157 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8158 }%
8159 \renewcommand*\Glsxtrinlinefullformat}[2]{%
8160 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8161 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8162 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
8163 }%
8164 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
8165 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8166 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8167 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8168 }%

```

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.

```

8169 \renewcommand*\glxtrfullformat}[2]{%
8170 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8171 \ifglxtrinsertinside\else##2\fi
8172 }%
8173 \renewcommand*\glxtrfullplformat}[2]{%
8174 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8175 \ifglxtrinsertinside\else##2\fi
8176 }%
8177 \renewcommand*\Glsxtrfullformat}[2]{%
8178 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8179 \ifglxtrinsertinside\else##2\fi
8180 }%
8181 \renewcommand*\Glsxtrfullplformat}[2]{%
8182 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8183 \ifglxtrinsertinside\else##2\fi
8184 }%
8185 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

8186 \newabbreviationstyle{long-em-noshort-em-noreg}%
8187 {%
8188   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%
      Unset the regular attribute if it has been set.
8189   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8190     \glshasattribute{\the\glslabeltok}{regular}%
8191     {%
8192       \glssetattribute{\the\glslabeltok}{regular}{false}%
8193     }%
8194   }%
8195 }%
8196 }%
8197 {%
8198   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
8199 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

8200 \newabbreviationstyle{long-noshort-em-desc}%
8201 {%
8202   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8203 }%
8204 {%
8205   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8206   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8207   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8208   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8209   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8210   \renewcommand*{\glsxtrsubsequentfmt}[2]{%
8211     \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8212     \ifglsxtrininsertinside \else##2\fi
8213   }%
8214   \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
8215     \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8216     \ifglsxtrininsertinside \else##2\fi
8217   }%
8218   \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8219     \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8220     \ifglsxtrininsertinside \else##2\fi
8221   }%
8222   \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8223     \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8224     \ifglsxtrininsertinside \else##2\fi
8225   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8226   \renewcommand*{\glsxtrinlinefullformat}[2]{%

```

```

8227 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8228 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8229 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8230 }%
8231 \renewcommand*{\glsxtrinelinefullplformat}[2]{%
8232 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8233 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8234 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8235 }%
8236 \renewcommand*{\Glsxtrinelinefullformat}[2]{%
8237 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8238 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8239 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8240 }%
8241 \renewcommand*{\Glsxtrinelinefullplformat}[2]{%
8242 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8243 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8244 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8245 }%

```

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.

```

8246 \renewcommand*{\glsxtrfullformat}[2]{%
8247 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8248 \ifglsxtrinsertinside\else##2\fi
8249 }%
8250 \renewcommand*{\glsxtrfullplformat}[2]{%
8251 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8252 \ifglsxtrinsertinside\else##2\fi
8253 }%
8254 \renewcommand*{\Glsxtrfullformat}[2]{%
8255 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8256 \ifglsxtrinsertinside\else##2\fi
8257 }%
8258 \renewcommand*{\Glsxtrfullplformat}[2]{%
8259 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8260 \ifglsxtrinsertinside\else##2\fi
8261 }%
8262 }

```

long-desc-em Backward compatibility:

```
8263 \@glsxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}
```

noshort-em-desc The short form is explicitly invoked through commands like `\glsshort`. The long form is emphasized.

```

8264 \newabbreviationstyle{long-em-noshort-em-desc}%
8265 {%
8266 \renewcommand*{\CustomAbbreviationFields}{%
8267 name={\protect\protect\glslongemfont{\the\glslongtok}},

```



```

8268     sort={\the\glslongtok},
8269     first={\protect\glsfirstlongemfont{\the\glslongtok}},
8270     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
8271     text={\glslongemfont{\the\glslongtok}},
8272     plural={\glslongemfont{\the\glslongpltok}}%
8273 }%
8274 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8275   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8276 }%
8277 {%
8278 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8279 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8280 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8281 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8282 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8283 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
8284   \glslongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8285   \ifglsxtrininsertinside \else##2\fi
8286 }%
8287 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
8288   \glslongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8289   \ifglsxtrininsertinside \else##2\fi
8290 }%
8291 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8292   \glslongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8293   \ifglsxtrininsertinside \else##2\fi
8294 }%
8295 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8296   \glslongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8297   \ifglsxtrininsertinside \else##2\fi
8298 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8299 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8300   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8301   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8302   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8303 }%
8304 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8305   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8306   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8307   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8308 }%
8309 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8310   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8311   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8312   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8313 }%

```

```

8314 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
8315   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8316   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8317   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8318 }%

```

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.

```

8319 \renewcommand*\glsxtrfullformat}[2]{%
8320   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8321   \ifglxtrinsertinside\else##2\fi
8322 }%
8323 \renewcommand*\glsxtrfullplformat}[2]{%
8324   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8325   \ifglxtrinsertinside\else##2\fi
8326 }%
8327 \renewcommand*\Glsxtrfullformat}[2]{%
8328   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8329   \ifglxtrinsertinside\else##2\fi
8330 }%
8331 \renewcommand*\Glsxtrfullplformat}[2]{%
8332   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8333   \ifglxtrinsertinside\else##2\fi
8334 }%
8335 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

8336 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
8337 {%
8338   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%

```

Unset the regular attribute if it has been set.

```

8339 \renewcommand*\GlsXtrPostNewAbbreviation}{%
8340   \glshasattribute{\the\glslabeltok}{regular}%
8341   {%
8342     \glssetattribute{\the\glslabeltok}{regular}{false}%
8343     }%
8344   }%
8345 }%
8346 }%
8347 {%
8348   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
8349 }

```

ort-em-footnote

```

8350 \newabbreviationstyle{short-em-footnote}%
8351 {%
8352   \renewcommand*\CustomAbbreviationFields}{%
8353     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8354     sort={\the\glsshorttok},

```

```

8355 description={\the\glslongtok},%
8356 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8357 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8358 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8359 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8360 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8361 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8362 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

8363 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8364 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8365 \glsattribute{\the\glslabeltok}{regular}%
8366 {%
8367 \glssetattribute{\the\glslabeltok}{regular}{false}%
8368 }%
8369 {}%
8370 }%
8371 }%
8372 {%

```

```

8373 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8374 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8375 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8376 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
8377 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

8378 \renewcommand*{\glxtrfullformat}[2]{%
8379 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8380 \ifglxtrinsertinside\else##2\fi
8381 \protect\glxtrabbrvfootnote{##1}%
8382 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8383 }%
8384 \renewcommand*{\glxtrfullplformat}[2]{%
8385 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8386 \ifglxtrinsertinside\else##2\fi
8387 \protect\glxtrabbrvfootnote{##1}%
8388 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8389 }%
8390 \renewcommand*{\Glsxtrfullformat}[2]{%
8391 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8392 \ifglxtrinsertinside\else##2\fi
8393 \protect\glxtrabbrvfootnote{##1}%
8394 {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8395 }%
8396 \renewcommand*{\Glsxtrfullplformat}[2]{%
8397 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8398 \ifglxtrinsertinside\else##2\fi
8399 \protect\glxtrabbrvfootnote{##1}%

```

```

8400     {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8401 }%

The first use full form and the inline full form use the short (long) style.

8402 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8403   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8404   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8405   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8406 }%
8407 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8408   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8409   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8410   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8411 }%
8412 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8413   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8414   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8415   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8416 }%
8417 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8418   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8419   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8420   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8421 }%
8422 }

```

footnote-em Backward compatibility:

```
8423 \@glsxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}
```

em-postfootnote

```

8424 \newabbreviationstyle{short-em-postfootnote}%
8425 {%
8426   \renewcommand*{\CustomAbbreviationFields}{%
8427     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8428     sort={\the\glsshorttok},
8429     description={\the\glslongtok},%
8430     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},%
8431     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},%
8432     plural={\protect\glsabbrvemfont{\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.

```

8433 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8434   \csdef{glsxtrpostlink\glscategorylabel}{%
8435     \glsxtrifwasfirstuse
8436     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
8437     \glsxtrdropostpunc{\protect\glsxtrabbrvfootnote{\glslabel}}%
```

```

8438     {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%
8439   }%
8440   {}%
8441   }%
8442   \glsattribute{\the\glslabeltok}{regular}%
8443   {%
8444   \glssetattribute{\the\glslabeltok}{regular}{false}%
8445   }%
8446   {}%
8447   }%

```

The footnote needs to be suppressed in the inline form, so `\glsxtrfull` must set the first use switch off.

```

8448   \renewcommand*\glsxtrsetupfulldefs{%
8449     \let\glsxtrifwasfirstuse\@secondoftwo
8450   }%
8451 }%
8452 {%
8453   \renewcommand*\abbrvpluralsuffix{\protect\glsxtremsuffix}%
8454   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
8455   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
8456   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8457   \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8458   \renewcommand*\glsxtrfullformat[2]{%
8459     \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8460     \ifglsxtrininsertinside\else##2\fi
8461   }%
8462   \renewcommand*\glsxtrfullplformat[2]{%
8463     \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8464     \ifglsxtrininsertinside\else##2\fi
8465   }%
8466   \renewcommand*\Glsxtrfullformat[2]{%
8467     \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8468     \ifglsxtrininsertinside\else##2\fi
8469   }%
8470   \renewcommand*\Glsxtrfullplformat[2]{%
8471     \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8472     \ifglsxtrininsertinside\else##2\fi
8473   }%

```

The first use full form and the inline full form use the short (long) style.

```

8474   \renewcommand*\glsxtrininlinefullformat[2]{%
8475     \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8476     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8477     \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8478   }%
8479   \renewcommand*\glsxtrininlinefullplformat[2]{%
8480     \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8481     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%

```

```

8482   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8483 }%
8484 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8485   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
8486   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8487   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8488 }%
8489 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8490   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
8491   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8492   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8493 }%
8494 }

```

postfootnote-em Backward compatibility:

```
8495 \@glsxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}
```

1.6.6 Predefined Styles (User Parentheses Hook)

These styles allow the user to adjust the parenthetical forms. These styles all test for the existence of the field given by:

glsxtruserfield Default is the useri field.

```
8496 \newcommand*{\glsxtruserfield}{useri}
```

glsxtruserparen The format of the parenthetical information. The first argument is the long/short form. The second argument is the entry's label. If `\glscurrentfieldvalue` has been defined, then we have at least glossaries v4.23, which makes it easier for the user to adjust this.

```

8497 \ifdef\glscurrentfieldvalue
8498 {
8499   \newcommand*{\glsxtruserparen}[2]{%
8500     \glsxtrfullsep{#2}%
8501     \glsxtrparen
8502     {#1\ifglshasfield{\glsxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
8503   }
8504 }
8505 {
8506   \newcommand*{\glsxtruserparen}[2]{%
8507     \glsxtrfullsep{#2}%
8508     \glsxtrparen
8509     {#1\ifglshasfield{\glsxtruserfield}{#2}{, \@glo@thisvalue}{}}%
8510   }
8511 }

```

Font used for short form:

lsabbrvuserfont

```
8512 \newcommand*{\glsabbrvuserfont}[1]{\glsabbrvdefaultfont{#1}}
```

Font used for short form on first use:

stabbrvuserfont

```
8513 \newcommand*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}
```

Font used for long form:

glslonguserfont

```
8514 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}
```

Font used for long form on first use:

rstlonguserfont

```
8515 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
8516 \newcommand*{\glsxtrusersuffix}{\glsxtrabbrvpluralsuffix}
```

long-short-user

```
8517 \newabbreviationstyle{long-short-user}%
```

```
8518 {%
```

```
8519   \renewcommand*{CustomAbbreviationFields}{%
```

```
8520     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
```

```
8521     sort={\the\glsshorttok},
```

```
8522     first={\protect\glsfirstlonguserfont{\the\glslongtok}}%
```

```
8523     \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
```

```
8524     {\the\glslabeltok}},%
```

```
8525     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}}%
```

```
8526     \protect\glsxtruserparen
```

```
8527     {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
```

```
8528     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
```

```
8529     description={\protect\glslonguserfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
8530 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
8531   \glshasattribute{\the\glslabeltok}{regular}}%
```

```
8532   {%
```

```
8533     \glissetattribute{\the\glslabeltok}{regular}{false}}%
```

```
8534   }%
```

```
8535   {}%
```

```
8536 }%
```

```
8537 }%
```

```
8538 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
8539 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
```

```
8540 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
```

```
8541 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
```

```
8542 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
```

```
8543 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```

8544 \renewcommand*{\glxtrfullformat}[2]{%
8545   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8546   \ifglxtrinsertinside\else##2\fi
8547   \glxtruserparen{\glsfirstabbruserfont{\glsaccessshort{##1}}}{##1}%
8548 }%
8549 \renewcommand*{\glxtrfullplformat}[2]{%
8550   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8551   \ifglxtrinsertinside\else##2\fi
8552   \glxtruserparen{\glsfirstabbruserfont{\glsaccessshortpl{##1}}}{##1}%
8553 }%
8554 \renewcommand*{\Glsxtrfullformat}[2]{%
8555   \Glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8556   \ifglxtrinsertinside\else##2\fi
8557   \glxtruserparen{\Glsfirstabbruserfont{\Glsaccessshort{##1}}}{##1}%
8558 }%
8559 \renewcommand*{\Glsxtrfullplformat}[2]{%
8560   \Glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8561   \ifglxtrinsertinside\else##2\fi
8562   \glxtruserparen{\Glsfirstabbruserfont{\Glsaccessshortpl{##1}}}{##1}%
8563 }%
8564 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

8565 \newabbreviationstyle{long-postshort-user}%
8566 {%
8567   \renewcommand*{\CustomAbbreviationFields}{%
8568     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
8569     sort={\the\glsshorttok},
8570     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
8571     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
8572     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
8573     description={\protect\glslonguserfont{\the\glslongtok}}}%
8574 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8575   \csdef{glxtrpostlink\glscategorylabel}{%
8576     \glxtrifwasfirstuse
8577     {%
8578       \glxtruserparen
8579       {\glsfirstabbruserfont{\glsentryshort{\glslabel}}}%
8580       {\glslabel}%
8581     }%
8582   }%
8583 }%
8584 \glsattribute{\the\glslabeltok}{regular}%
8585 {%
8586   \glsattribute{\the\glslabeltok}{regular}{false}%
8587 }%
8588 }%

```



```

8589 }%
8590 }%
8591 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8592 \renewcommand*\abbrvpluralsuffix{\glsxtrusersuffix}%
8593 \renewcommand*\glsabbrvfont[1]{\glsabbrvuserfont{##1}}%
8594 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvuserfont{##1}}%
8595 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonguserfont{##1}}%
8596 \renewcommand*\glslongfont[1]{\glslonguserfont{##1}}%

```

First use full form:

```

8597 \renewcommand*\glsxtrfullformat[2]{%
8598   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8599   \ifglsxtrininsertinside\else##2\fi
8600 }%
8601 \renewcommand*\glsxtrfullplformat[2]{%
8602   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8603   \ifglsxtrininsertinside\else##2\fi
8604 }%
8605 \renewcommand*\Glsxtrfullformat[2]{%
8606   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8607   \ifglsxtrininsertinside\else##2\fi
8608 }%
8609 \renewcommand*\Glsxtrfullplformat[2]{%
8610   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8611   \ifglsxtrininsertinside\else##2\fi
8612 }%

```

In-line format:

```

8613 \renewcommand*\glsxtrinlinefullformat[2]{%
8614   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8615   \ifglsxtrininsertinside\else##2\fi
8616   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
8617 }%
8618 \renewcommand*\glsxtrinlinefullplformat[2]{%
8619   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8620   \ifglsxtrininsertinside\else##2\fi
8621   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
8622 }%
8623 \renewcommand*\Glsxtrinlinefullformat[2]{%
8624   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8625   \ifglsxtrininsertinside\else##2\fi
8626   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
8627 }%
8628 \renewcommand*\Glsxtrinlinefullplformat[2]{%
8629   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8630   \ifglsxtrininsertinside\else##2\fi
8631   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
8632 }%
8633 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```
8634 \newabbreviationstyle{long-postshort-user}%
8635 {%
8636   \renewcommand*{\CustomAbbreviationFields}{%
8637     name={\protect\glslonguserfont{\the\glslongtok}}%
8638     \protect\glsxtruserparen
8639     {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}},
8640   sort={\the\glslongtok},
8641   first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
8642   firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

8643   text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
8644   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
8645 }%
8646 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8647   \csdef{glsxtrpostlink\glscategorylabel}{%
8648     \glsxtrifwasfirstuse
8649     {%
8650       \glsxtruserparen
8651       {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
8652       {\glslabel}}%
8653   }%
8654   {%
8655   }%
8656   \glsattribute{\the\glslabeltok}{regular}%
8657   {%
8658     \glssetattribute{\the\glslabeltok}{regular}{false}%
8659   }%
8660   {}}%
8661 }%
8662 }%
8663 {%
8664   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
8665 }
```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```
8666 \newabbreviationstyle{short-postlong-user}%
8667 {%
8668   \renewcommand*{\CustomAbbreviationFields}{%
8669     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
8670     sort={\the\glsshorttok},
8671     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
8672     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

8673     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
8674     description={\protect\glslonguserfont{\the\glslongtok}}}%
8675 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8676   \csdef{glsxtrpostlink\glscategorylabel}{%
8677     \glsxtrifwasfirstuse
8678     {%
```

```

8679     \glxtruserparen
8680         {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
8681         {\glslabel}%
8682     }%
8683     {}%
8684 }%
8685 \glshasattribute{\the\glslabeltok}{regular}%
8686 {%
8687     \glsssetattribute{\the\glslabeltok}{regular}{false}%
8688 }%
8689 {}%
8690 }%
8691 }%
8692 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8693 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
8694 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
8695 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
8696 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
8697 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

8698 \renewcommand*{\glxtrfullformat}[2]{%
8699     \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8700     \ifglxtrininsertinside\else##2\fi
8701 }%
8702 \renewcommand*{\glxtrfullplformat}[2]{%
8703     \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8704     \ifglxtrininsertinside\else##2\fi
8705 }%
8706 \renewcommand*{\Glsxtrfullformat}[2]{%
8707     \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8708     \ifglxtrininsertinside\else##2\fi
8709 }%
8710 \renewcommand*{\Glsxtrfullplformat}[2]{%
8711     \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8712     \ifglxtrininsertinside\else##2\fi
8713 }%

```

In-line format:

```

8714 \renewcommand*{\glxtrinlinefullformat}[2]{%
8715     \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8716     \ifglxtrininsertinside\else##2\fi
8717     \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
8718 }%
8719 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8720     \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8721     \ifglxtrininsertinside\else##2\fi
8722     \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%

```

```

8723 }%
8724 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8725   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8726   \ifglsxtrininsertinside\else##2\fi
8727   \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
8728 }%
8729 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8730   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8731   \ifglsxtrininsertinside\else##2\fi
8732   \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
8733 }%
8734 }

```

`long-user-desc` Like `short-postlong-user` but leaves the user to specify the description.

```

8735 \newabbreviationstyle{short-postlong-user-desc}%
8736 {%
8737   \renewcommand*{\CustomAbbreviationFields}{%
8738     name={\protect\glsabbrvuserfont{\the\glsshorttok}}%
8739     \protect\glsxtruserparen
8740       {\protect\glslonguserfont{\the\glslongpltok}}%
8741       {\the\glslabeltok}},
8742   sort={\the\glsshorttok},
8743   first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
8744   firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
8745   text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
8746   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
8747 }%
8748 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8749   \csdef{glsxtrpostlink\glscategorylabel}{%
8750     \glsxtrifwasfirstuse
8751     {%
8752       \glsxtruserparen
8753         {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
8754         {\glslabel}}%
8755     }%
8756   }%
8757 }%
8758 \glsattribute{\the\glslabeltok}{regular}%
8759 {%
8760   \glssetattribute{\the\glslabeltok}{regular}{false}%
8761 }%
8762 }%
8763 }%
8764 }%
8765 {%
8766   \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
8767 }

```

short-user-desc

```
8768 \newabbreviationstyle{long-short-user-desc}%
8769 {%
8770   \renewcommand*{\CustomAbbreviationFields}{%
8771     name={\glxtrlongshortdescname},
8772     sort={\glxtrlongshortdescsort},%
8773     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
8774       \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
8775       {\the\glslabeltok}},%
8776     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
8777       \protect\glxtruserparen
8778       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
8779     text={\protect\glsabbrvfont{\the\glsshorttok}},%
8780     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
8781   }%
```

Unset the regular attribute if it has been set.

```
8782   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8783     \glshasattribute{\the\glslabeltok}{regular}%
8784     {%
8785       \glissetattribute{\the\glslabeltok}{regular}{false}%
8786       }%
8787     {}}%
8788   }%
8789 }%
8790 {%
8791   \GlsXtrUseAbbrStyleFmts{long-short-user}%
8792 }
```

short-long-user

```
8793 \newabbreviationstyle{short-long-user}%
8794 {%
8795   \glslonguserfont is used in the description since \glsdesc doesn't set the style.
8795   \renewcommand*{\CustomAbbreviationFields}{%
8796     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
8797     sort={\the\glsshorttok},
8798     description={\protect\glslonguserfont{\the\glslongtok}},%
8799     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
8800       \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
8801       {\the\glslabeltok}},%
8802     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
8803       \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%
8804       {\the\glslabeltok}},%
8805     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
8806   }%
```

Unset the regular attribute if it has been set.

```
8806   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8807     \glshasattribute{\the\glslabeltok}{regular}%
8808     {%
8809       \glissetattribute{\the\glslabeltok}{regular}{false}%
8810       }%
8811     {}}%
8812   }%
```

```

8807 \glshasattribute{\the\glslabeltok}{regular}%
8808 {%
8809 \glssetattribute{\the\glslabeltok}{regular}{false}%
8810 }%
8811 {}%
8812 }%
8813 }%
8814 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8815 \renewcommand*\abbrevpluralsuffix{\glsxtrusersuffix}%
8816 \renewcommand*\glsabbrvfont[1]{\glsabbrvuserfont{##1}}%
8817 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvuserfont{##1}}%
8818 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonguserfont{##1}}%
8819 \renewcommand*\glslongfont[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8820 \renewcommand*\glsxtrfullformat[2]{%
8821 \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8822 \ifglsxtrininsertinside\else##2\fi
8823 \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
8824 }%
8825 \renewcommand*\glsxtrfullplformat[2]{%
8826 \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8827 \ifglsxtrininsertinside\else##2\fi
8828 \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
8829 }%
8830 \renewcommand*\Glsxtrfullformat[2]{%
8831 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8832 \ifglsxtrininsertinside\else##2\fi
8833 \glsxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
8834 }%
8835 \renewcommand*\Glsxtrfullplformat[2]{%
8836 \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8837 \ifglsxtrininsertinside\else##2\fi
8838 \glsxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
8839 }%
8840 }

```

-long-user-desc

```

8841 \newabbreviationstyle{short-long-user-desc}%
8842 {%
8843 \renewcommand*\CustomAbbreviationFields{%
8844 name={\glsxtrshortlongdescname},
8845 sort={\glsxtrshortlongdescsort},%
8846 first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
8847 \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
8848 {\the\glslabeltok}},%
8849 firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%

```

```

8850   \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
8851   {\the\glslabeltok}},%
8852   text={\protect\glsabbrvfont{\the\glsshorttok}},%
8853   plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
8854 }%

```

Unset the regular attribute if it has been set.

```

8855 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8856   \glshasattribute{\the\glslabeltok}{regular}}%
8857   {%
8858   \glissetattribute{\the\glslabeltok}{regular}{false}}%
8859   }%
8860   {}%
8861 }%
8862 }%
8863 {%
8864 \GlsXtrUseAbbrStyleFmts{short-long-user}}%
8865 }

```

1.6.7 Predefined Styles (Hyphen)

These styles are designed to work with the `markwords` 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 `\glxtrlong` set `\glsinsert` to empty with the entire link-text stored in `\glscustomtext`.

`trifhyphenstart` Checks if the argument starts with a hyphen. The argument may be `\glsinsert` so check for that and expand.

```

8866 \newrobustcmd*{\glxtrifhyphenstart}[3]{%
8867   \ifx\glsinsert#1\relax
8868   \expandafter\@glxtrifhyphenstart#1\relax\relax
8869   \@end@glxtrifhyphenstart{#2}{#3}}%
8870 \else
8871   \@glxtrifhyphenstart#1\relax\relax\@end@glxtrifhyphenstart{#2}{#3}}%
8872 \fi
8873 }

```

`trifhyphenstart`

```

8874 \def\@glxtrifhyphenstart#1#2\@end@glxtrifhyphenstart#3#4{%
8875   \ifx-#1\relax#3\else #4\fi
8876 }

```

`rlonghyphenshort`

```
\glxtrlonghyphenshort{<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.

8877 \newcommand*{\glxtrlonghyphenshort}[4]{%

Grouping is needed to localise the redefinitions.

8878 {%

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.) No change is made to \glxtrwordsep if *<insert>* doesn't start with a hyphen.

8879 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{%

8880 \glxtriflonghyphenfont{#2\ifglxtrininsertinside{#4}\fi}%

8881 \ifglxtrininsertinside\else{#4}\fi

8882 \glxtrfullsep{#1}%

8883 \glxtrparen{\glxtrifabbrvhyphenfont{#3\ifglxtrininsertinside{#4}\fi}%

8884 \ifglxtrininsertinside\else{#4}\fi}%

8885 }%

8886 }

abbrvhyphenfont

8887 \newcommand*{\glsabbrvhyphenfont}{\glsabbrvdefaultfont}%

abbrvhyphenfont

8888 \newcommand*{\glsfirstabbrvhyphenfont}{\glsabbrvhyphenfont}%

slonghyphenfont

8889 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%

tlonghyphenfont

8890 \newcommand*{\glsfirstlonghyphenfont}{\glslonghyphenfont}%

The default short form suffix:

xtrhyphensuffix

8891 \newcommand*{\glxtrhyphensuffix}{\glxtrabbrvpluralsuffix}

en-short-hyphen Designed for use with the markwords attribute.

8892 \newabbreviationstyle{long-hyphen-short-hyphen}%

8893 {%

8894 \renewcommand*{\CustomAbbreviationFields}{%

8895 name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},

8896 sort={\the\glsshorttok},

8897 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%

8898 \protect\glxtrfullsep{\the\glslabeltok}}%

8899 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%

8900 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%

8901 \protect\glxtrfullsep{\the\glslabeltok}}%

8902 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%

8903 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}},%

8904 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

Unset the regular attribute if it has been set.

```
8905 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8906   \glsattribute{\the\glslabeltok}{regular}%
8907   {%
8908     \glssetattribute{\the\glslabeltok}{regular}{false}%
8909   }%
8910   {}%
8911 }%
8912 }%
8913 {%
8914 \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
8915 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
8916 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
8917 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
8918 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8919 \renewcommand*{\glsxtrfullformat}[2]{%
8920   \glsxtrlonghyphenshort{##1}{\glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
8921 }%
8922 \renewcommand*{\glsxtrfullplformat}[2]{%
8923   \glsxtrlonghyphenshort{##1}{\glsaccesslongpl{##1}}%
8924   {\glsaccessshortpl{##1}}{##2}%
8925 }%
8926 \renewcommand*{\Glsxtrfullformat}[2]{%
8927   \glsxtrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
8928 }%
8929 \renewcommand*{\Glsxtrfullplformat}[2]{%
8930   \glsxtrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
8931   {\glsaccessshortpl{##1}}{##2}%
8932 }%
8933 }
```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```
8934 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
8935 {%
8936   \renewcommand*{\CustomAbbreviationFields}{%
8937     name={\glsxtrlongshortdescname},
8938     sort={\glsxtrlongshortdescsort},
8939     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
8940       \protect\glsxtrfullsep{\the\glslabeltok}%
8941       \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
8942     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
8943       \protect\glsxtrfullsep{\the\glslabeltok}%
8944       \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
8945     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
8946     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
8947   }%
```

Unset the regular attribute if it has been set.

```

8948 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8949   \glshasattribute{\the\glslabeltok}{regular}%
8950   {%
8951     \glissetattribute{\the\glslabeltok}{regular}{false}%
8952   }%
8953   {}%
8954 }%
8955 }%
8956 {%
8957   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
8958 }

```

onghyphennoshort

```
\glsxtrlonghyphennoshort{<label>}{<long>}{<insert>}
```

```
8959 \newcommand*{\glsxtrlonghyphennoshort}[3]{%
```

Grouping is needed to localise the redefinitions.

```
8960 {%
```

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.

```

8961   \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
8962   \glsfirstlonghyphenfont{#2\ifglsxtrininsertinside{#3}\fi}%
8963   \ifglsxtrininsertinside\else{#3}\fi
8964 }%
8965 }

```

short-desc-noreg

This version doesn't show the short form (except explicitly with `\glsxtrshort`). Since `\glsxtrshort` 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.

```

8966 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
8967 {%
8968   \renewcommand*{\CustomAbbreviationFields}{%
8969     name={\protect\protect\glslonghyphenfont{\the\glslongtok}},
8970     sort={\expandonce\glsxtrorglong},
8971     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
8972     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
8973     plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
8974   }%

```

Unset the regular attribute if it has been set.

```

8975 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8976   \glshasattribute{\the\glslabeltok}{regular}%
8977   {%
8978     \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

8979   }%
8980   {}%
8981  }%
8982 }%
8983 {%
8984  \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8985  \renewcommand*\abbrvpluralsuffix{\glsxtrabbrvpluralsuffix}%
8986  \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
8987  \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
8988  \renewcommand*\glsfirstlongfont[1]{\glsfirstlonghyphenfont{##1}}%
8989  \renewcommand*\glslongfont[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8990  \renewcommand*\glsxtrsubsequentfmt[2]{%
8991    \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
8992  }%
8993  \renewcommand*\glsxtrsubsequentplfmt[2]{%
8994    \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
8995  }%
8996  \renewcommand*\Glsxtrsubsequentfmt[2]{%
8997    \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
8998  }%
8999  \renewcommand*\Glsxtrsubsequentplfmt[2]{%
9000    \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9001  }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9002  \renewcommand*\glsxtrinilinefullformat[2]{%
9003    \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9004    \glsxtrfullsep{##1}%
9005    \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9006  }%
9007  \renewcommand*\glsxtrinilinefullplformat[2]{%
9008    \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9009    \glsxtrfullsep{##1}%
9010    \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
9011  }%
9012  \renewcommand*\Glsxtrinilinefullformat[2]{%
9013    \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9014    \glsxtrfullsep{##1}%
9015    \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9016  }%
9017  \renewcommand*\Glsxtrinilinefullplformat[2]{%
9018    \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9019    \glsxtrfullsep{##1}%
9020    \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
9021  }%

```

The first use full form only displays the long form.

```

9022 \renewcommand*\glxtrfullformat}[2]{%
9023   \glxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9024 }%
9025 \renewcommand*\glxtrfullplformat}[2]{%
9026   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9027 }%
9028 \renewcommand*\Glsxtrfullformat}[2]{%
9029   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9030 }%
9031 \renewcommand*\Glsxtrfullplformat}[2]{%
9032   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9033 }%
9034 }

```

n-noshort-noreg It doesn't really make a great deal of sense to have a long-only style that doesn't have a descriptionless 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.

```

9035 \newabbreviationstyle{long-hyphen-noshort-noreg}%
9036 {%
9037   \renewcommand*\CustomAbbreviationFields{%
9038     name={\protect\glsabbrvfont{\the\glsshorttok}},
9039     sort={\the\glsshorttok},
9040     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9041     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9042     text={\protect\glslonghyphenfont{\the\glslongtok}},%
9043     plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
9044     description={\the\glslongtok}%
9045   }%

```

Unset the regular attribute if it has been set.

```

9046 \renewcommand*\GlsXtrPostNewAbbreviation{%
9047   \glshasattribute{\the\glslabeltok}{regular}%
9048   {%
9049     \glissetattribute{\the\glslabeltok}{regular}{false}%
9050   }%
9051   {}%
9052 }%
9053 }%
9054 {%
9055   \GlsXtrUseAbbrStyleFmts{long-desc}%
9056 }

```

glxtrlonghyphen `\glxtrlonghyphen{<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.

```

9057 \newcommand*\glxtrlonghyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

9058 {%
9059   \glstrifhyphenstart{#3}{\def\glstrwordsep{-}}{ }%
9060   \glfirstlonghyphenfont{#1}%
9061 }%
9062 }

```

posthyphenshort `\glstrposthyphenshort{<label>}{<insert>}`

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like `\glstrlonghyphenshort` but omits the *<long>* part. This always uses the singular short form.

```

9063 \newcommand*{\glstrposthyphenshort}[2]{%
9064   {%
9065     \glstrifhyphenstart{#2}{\def\glstrwordsep{-}}{ }%
9066     \ifglstrinsertinside{\glfirstlonghyphenfont{#2}}\else{#2}\fi
9067     \glstrfullsep{#1}%
9068     \glstrparen
9069     {\glfirstabbrvhyphenfont{\glentryshort{#1}}\ifglstrinsertinside{#2}\fi}%
9070     \ifglstrinsertinside\else{#2}\fi
9071   }%
9072 }%
9073 }

```

hyphensubsequent `\glstrposthyphensubsequent{<label>}{<insert>}`

Format in the post-link hook for subsequent use. The label is ignored by default.

```

9074 \newcommand*{\glstrposthyphensubsequent}[2]{%
9075   \glabbrvfont{\ifglstrinsertinside {#2}\fi}%
9076   \ifglstrinsertinside \else{#2}\fi
9077 }

```

postshort-hyphen Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

9078 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
9079 {%
9080   \renewcommand*{\CustomAbbreviationFields}{%
9081     name={\protect\glabbrvhyphenfont{\the\glsshorttok}},
9082     sort={\the\glsshorttok},
9083     first={\protect\glfirstlonghyphenfont{\the\glslongtok}},%
9084     firstplural={\protect\glfirstlonghyphenfont{\the\glslongpltok}},%
9085     plural={\protect\glabbrvhyphenfont{\the\glshortpltok}},%
9086     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
9087   \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

9088 \csdef{glxtrpostlink\glscategorylabel}{%
9089 \glxtrifwasfirstuse
9090 {%
9091 \glxtrposthyphenshort{\glslabel}{\glinsert}%
9092 }%
9093 {%

```

Put the insertion into the post-link:

```

9094 \glxtrposthyphensubsequent{\glslabel}{\glinsert}%
9095 }%
9096 }%
9097 \glshasattribute{\the\glslabeltok}{regular}%
9098 {%
9099 \glsetattribute{\the\glslabeltok}{regular}{false}%
9100 }%
9101 {}%
9102 }%
9103 }%
9104 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9105 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
9106 \renewcommand*{\glabbrvfont}[1]{\glabbrvhyphenfont{##1}}%
9107 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvhyphenfont{##1}}%
9108 \renewcommand*{\glfirstlongfont}[1]{\glfirstlonghyphenfont{##1}}%
9109 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

9110 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9111 \glabbrvfont{\glaccessshort{##1}}%
9112 }%
9113 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9114 \glabbrvfont{\glaccessshortpl{##1}}%
9115 }%
9116 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9117 \glabbrvfont{\Glsaccessshort{##1}}%
9118 }%
9119 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9120 \glabbrvfont{\Glsaccessshortpl{##1}}%
9121 }%

```

First use full form:

```

9122 \renewcommand*{\glxtrfullformat}[2]{%
9123 \glxtrlonghyphen{\glaccesslong{##1}}{##1}{##2}%
9124 }%
9125 \renewcommand*{\glxtrfullplformat}[2]{%
9126 \glxtrlonghyphen{\glaccesslongpl{##1}}{##1}{##2}%
9127 }%
9128 \renewcommand*{\Glsxtrfullformat}[2]{%
9129 \glxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
9130 }%

```

```

9131 \renewcommand*\Glsxtrfullplformat}[2]{%
9132   \glsxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
9133 }%

```

In-line format.

```

9134 \renewcommand*\glsxtrinlinefullformat}[2]{%
9135   \glsfirstlonghyphenfont{\glsaccesslong{##1}%
9136     \ifglsxtrininsertinside{##2}\fi}%
9137   \ifglsxtrininsertinside \else{##2}\fi
9138 }%
9139 \renewcommand*\glsxtrinlinefullplformat}[2]{%
9140   \glsfirstlonghyphenfont{\glsaccesslongpl{##1}%
9141     \ifglsxtrininsertinside{##2}\fi}%
9142   \ifglsxtrininsertinside \else{##2}\fi
9143 }%
9144 \renewcommand*\Glsxtrinlinefullformat}[2]{%
9145   \glsfirstlonghyphenfont{\Glsaccesslong{##1}%
9146     \ifglsxtrininsertinside{##2}\fi}%
9147   \ifglsxtrininsertinside \else{##2}\fi
9148 }%
9149 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
9150   \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
9151     \ifglsxtrininsertinside{##2}\fi}%
9152   \ifglsxtrininsertinside \else{##2}\fi
9153 }%
9154 }

```

ort-hyphen-desc Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

9155 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
9156 {%
9157   \renewcommand*\CustomAbbreviationFields{%
9158     name={\glsxtrlongshortdescname},
9159     sort={\glsxtrlongshortdescsort},%
9160     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9161     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9162     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
9163     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
9164   }%
9165   \renewcommand*\GlsXtrPostNewAbbreviation{%
9166     \csdef{glsxtrpostlink\glscategorylabel}{%
9167       \glsxtrifwasfirstuse
9168       {%
9169         \glsxtrposthyphenshort{\glslabel}{\glsinsert}%
9170       }%
9171     }%

```

Put the insertion into the post-link:

```

9172       \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
9173     }%
9174   }%

```

```

9175 \glshasattribute{\the\glslabeltok}{regular}%
9176 {%
9177 \glissetattribute{\the\glslabeltok}{regular}{false}%
9178 }%
9179 {}%
9180 }%
9181 }%
9182 {%
9183 \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
9184 }

```

rshorthyphenlong

```
\glsxtrshorthyphenlong{<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.

```
9185 \newcommand*{\glsxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
9186 {%
```

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.)

```

9187 \glxtrifhyphenstart{#4}{\def\glsxtrwordsep{-}}{}%
9188 \glsfirstabbrvhyphenfont{#2\ifglsxtrininsertinside{#4}\fi}%
9189 \ifglsxtrininsertinsideelse{#4}\fi
9190 \glsxtrfullsep{#1}%
9191 \glsxtrparen{\glsfirstlonghyphenfont{#3\ifglsxtrininsertinside{#4}\fi}%
9192 \ifglsxtrininsertinsideelse{#4}\fi}%
9193 }%
9194 }

```

hen-long-hyphen

Designed for use with the `markwords` attribute.

```

9195 \newabbreviationstyle{short-hyphen-long-hyphen}%
9196 {%
9197 \renewcommand*{\CustomAbbreviationFields}{%
9198 name={\protect\glssabbrvhyphenfont{\the\glssshorttok}},
9199 sort={\the\glssshorttok},
9200 first={\protect\glsfirstabbrvhyphenfont{\the\glssshorttok}}%
9201 \protect\glsxtrfullsep{\the\glslabeltok}}%
9202 \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
9203 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glssshortpltok}}%
9204 \protect\glsxtrfullsep{\the\glslabeltok}}%
9205 \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
9206 plural={\protect\glssabbrvhyphenfont{\the\glssshortpltok}}},%
9207 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```


Unset the regular attribute if it has been set.

```

9208 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9209   \glshasattribute{\the\glslabeltok}{regular}%
9210   {%
9211     \glissetattribute{\the\glslabeltok}{regular}{false}%
9212   }%
9213   {}%
9214 }%
9215 }%
9216 {%
9217 \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
9218 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9219 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9220 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9221 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9222 \renewcommand*{\glsxtrfullformat}[2]{%
9223   \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
9224 }%
9225 \renewcommand*{\glsxtrfullplformat}[2]{%
9226   \glsxtrshorthyphenlong{##1}%
9227   {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
9228 }%
9229 \renewcommand*{\Glsxtrfullformat}[2]{%
9230   \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
9231 }%
9232 \renewcommand*{\Glsxtrfullplformat}[2]{%
9233   \glsxtrshorthyphenlong{##1}%
9234   {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
9235 }%
9236 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

9237 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
9238 {%
9239   \renewcommand*{\CustomAbbreviationFields}{%
9240     name={\glsxtrshortlongdescname},
9241     sort={\glsxtrshortlongdescsort},
9242     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%
9243     \protect\glsxtrfullsep{\the\glslabeltok}%
9244     \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
9245     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%
9246     \protect\glsxtrfullsep{\the\glslabeltok}%
9247     \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
9248     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
9249     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
9250   }%

```

Unset the regular attribute if it has been set.

```

9251 \renewcommand*\GlsXtrPostNewAbbreviation}{%
9252   \glshasattribute{\the\glslabeltok}{regular}%
9253   {%
9254     \glissetattribute{\the\glslabeltok}{regular}{false}%
9255   }%
9256   {}%
9257 }%
9258 }%
9259 {%
9260   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
9261 }

```

lsxtrshorthyphen

```
\glsxtrshorthyphen{<short>}{<label>}{<insert>}
```

Used by short-hyphen-postlong-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.

```

9262 \newcommand*\glsxtrshorthyphen}[3]{%
    Grouping is needed to localise the redefinitions.
9263   {%
9264     \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
9265     \glsfirstabbrvhyphenfont{#1}%
9266   }%
9267 }

```

trposthyphenlong

```
\glsxtrposthyphenlong{<label>}{<insert>}
```

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glsxtrshorthyphenlong` but omits the *<short>* part. This always uses the singular long form.

```

9268 \newcommand*\glsxtrposthyphenlong}[2]{%
9269   {%
9270     \glsxtrifhyphenstart{#2}{\def\glsxtrwordsep{-}}{}%
9271     \ifglsxtrinertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
9272     \glsxtrfullsep{#1}%
9273     \glsxtrparen
9274     {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglsxtrinertinside{#2}\fi}%
9275     \ifglsxtrinertinside\else{#2}\fi
9276   }%
9277 }%
9278 }

```

postlong-hyphen

Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

9279 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
9280 {%
9281   \renewcommand*{\CustomAbbreviationFields}{%
9282     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
9283     sort={\the\glsshorttok},
9284     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
9285     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
9286     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
9287     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
9288 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9289   \csdef{glsxtrpostlink\glscategorylabel}{%
9290     \glsxtrifwasfirstuse
9291     {%
9292       \glsxtrposthyphenlong{\glslabel}{\glsinsert}%
9293     }%
9294   }%

```

Put the insertion into the post-link:

```

9295     \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
9296   }%
9297 }%
9298 \glsattribute{\the\glslabeltok}{regular}%
9299 {%
9300   \glssetattribute{\the\glslabeltok}{regular}{false}%
9301 }%
9302 {}%
9303 }%
9304 }%
9305 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9306 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
9307 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9308 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9309 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9310 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

9311 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9312   \glsabbrvfont{\glsaccessshort{##1}}%
9313 }%
9314 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9315   \glsabbrvfont{\glsaccessshortpl{##1}}%
9316 }%
9317 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9318   \glsabbrvfont{\Glsaccessshort{##1}}%
9319 }%
9320 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9321   \glsabbrvfont{\Glsaccessshortpl{##1}}%
9322 }%

```

First use full form:

```

9323 \renewcommand*{\glxtrfullformat}[2]{%
9324   \glxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
9325 }%
9326 \renewcommand*{\glxtrfullplformat}[2]{%
9327   \glxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
9328 }%
9329 \renewcommand*{\Glsxtrfullformat}[2]{%
9330   \glxtrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
9331 }%
9332 \renewcommand*{\Glsxtrfullplformat}[2]{%
9333   \glxtrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
9334 }%

```

In-line format. Commands like `\glxtrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

9335 \renewcommand*{\glxtrinlinefullformat}[2]{%
9336   \glsfirstabbrvhyphenfont{\glsaccessshort{##1}}%
9337   \ifglxtrininsertinside{##2}\fi}%
9338   \ifglxtrininsertinside \else{##2}\fi
9339 }%
9340 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9341   \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}}%
9342   \ifglxtrininsertinside{##2}\fi}%
9343   \ifglxtrininsertinside \else{##2}\fi
9344 }%
9345 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9346   \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}}%
9347   \ifglxtrininsertinside{##2}\fi}%
9348   \ifglxtrininsertinside \else{##2}\fi
9349 }%
9350 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9351   \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}}%
9352   \ifglxtrininsertinside{##2}\fi}%
9353   \ifglxtrininsertinside \else{##2}\fi
9354 }%
9355 }

```

`ong-hyphen-desc` Like `short-hyphen-postlong-hyphen` but the description must be supplied by the user.

```

9356 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
9357 {%
9358   \renewcommand*{\CustomAbbreviationFields}{%
9359     name={\glxtrshortlongdescname},
9360     sort={\glxtrshortlongdescsort},%
9361     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
9362     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
9363     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
9364     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
9365 }%

```

```

9366 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9367   \csdef{glsxtrpostlink\glscategorylabel}{%
9368     \glsxtrifwasfirstuse
9369     {%
9370       \glsxtrposthyphenlong{\glslabel}{\glsinsert}%
9371     }%
9372   {%

```

Put the insertion into the post-link:

```

9373     \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
9374   }%
9375 }%
9376 \glsattribute{\the\glslabeltok}{regular}%
9377 {%
9378   \glssetattribute{\the\glslabeltok}{regular}{false}%
9379 }%
9380 {}%
9381 }%
9382 }%
9383 {%
9384 \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
9385 }

```

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

```
9386 \newcommand*{\glsabbrvonlyfont}{\glsabbrvdefaultfont}%
```

stabbrvonlyfont

```
9387 \newcommand*{\glsfirstabbrvonlyfont}{\glsabbrvonlyfont}%
```

glslongonlyfont

```
9388 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%
```

rstlongonlyfont

```
9389 \newcommand*{\glsfirstlongonlyfont}{\glslongonlyfont}%
```

The default short form suffix:

lsxtronlysuffix

```
9390 \newcommand*{\glsxtronlysuffix}{\glsxtrabbrvpluralsuffix}
```

only-short-only

```

9391 \newabbreviationstyle{long-only-short-only}%
9392 {%
9393   \renewcommand*{\CustomAbbreviationFields}{%
9394     name={\protect\glsabbrvonlyfont{\the\glsshorttok}},

```

```

9395     sort={\the\glsshorttok},
9396     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
9397     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
9398     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
9399     description={\protect\glslongonlyfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

9400 \renewcommand*\GlsXtrPostNewAbbreviation{%
9401   \glsattribute{\the\glslabeltok}{regular}%
9402   {%
9403     \glssetattribute{\the\glslabeltok}{regular}{false}%
9404   }%
9405   {}}%
9406 }%
9407 }%
9408 {%
9409 \renewcommand*\abbrvpluralsuffix{\protect\glsxtronlysuffix}%
9410 \renewcommand*\glsabbrvfont[1]{\glsabbrvonlyfont{##1}}%
9411 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvonlyfont{##1}}%
9412 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongonlyfont{##1}}%
9413 \renewcommand*\glslongfont[1]{\glslongonlyfont{##1}}%

```

The first use full form doesn't show the short form.

```

9414 \renewcommand*\glsxtrfullformat[2]{%
9415   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9416   \ifglsxtrininsertinside\else##2\fi
9417 }%
9418 \renewcommand*\glsxtrfullplformat[2]{%
9419   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9420   \ifglsxtrininsertinside\else##2\fi
9421 }%
9422 \renewcommand*\Glsxtrfullformat[2]{%
9423   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9424   \ifglsxtrininsertinside\else##2\fi
9425 }%
9426 \renewcommand*\Glsxtrfullplformat[2]{%
9427   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9428   \ifglsxtrininsertinside\else##2\fi
9429 }%

```

The inline full form does show the short form.

```

9430 \renewcommand*\glsxtrinlinefullformat[2]{%
9431   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9432   \ifglsxtrininsertinside\else##2\fi
9433   \glsxtrfullsep{##1}%
9434   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
9435 }%
9436 \renewcommand*\glsxtrinlinefullplformat[2]{%
9437   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9438   \ifglsxtrininsertinside\else##2\fi
9439   \glsxtrfullsep{##1}%

```

```

9440   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
9441 }%
9442 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9443   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9444   \ifglsxtrininsertinside\else##2\fi
9445   \glsxtrfullsep{##1}%
9446   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
9447 }%
9448 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9449   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9450   \ifglsxtrininsertinside\else##2\fi
9451   \glsxtrfullsep{##1}%
9452   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
9453 }%
9454 }

```

xtronlydescsort

```

9455 \newcommand*{\glsxtronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

9456 \newcommand*{\glsxtronlydescname}{%
9457   \protect\glslongfont{\the\glslongtok}}%
9458 }

```

short-only-desc

```

9459 \newabbreviationstyle{long-only-short-only-desc}%
9460 {%
9461   \renewcommand*{\CustomAbbreviationFields}{%
9462     name={\glsxtronlydescname},
9463     sort={\glsxtronlydescsort},%
9464     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
9465     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
9466     text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%
9467     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}}}%
9468 }%

```

Unset the regular attribute if it has been set.

```

9469 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9470   \glshasattribute{\the\glslabeltok}{regular}}%
9471 {%
9472   \glssetattribute{\the\glslabeltok}{regular}{false}}%
9473 }%
9474 {}%
9475 }%
9476 }%
9477 {%
9478   \GlsXtrUseAbbrStyleFmts{long-only-short-only}}%
9479 }

```

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:

```
9480 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
9481 \renewcommand*{\markright}[1]{%
9482   \glsxtrmarkhook
9483   \@glsxtr@org@markright{\@glsxtrinmark#1\@glsxtrnotinmark}%
9484   \glsxtrrestoremarkhook
9485 }
```

`\markboth` Save original definition:

```
9486 \let\@glsxtr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
9487 \renewcommand*{\markboth}[2]{%
9488   \glsxtrmarkhook
9489   \@glsxtr@org@markboth
9490   {\@glsxtrinmark#1\@glsxtrnotinmark}%
9491   {\@glsxtrinmark#2\@glsxtrnotinmark}%
9492   \glsxtrrestoremarkhook
9493 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```
9494 \newcommand*{\glxtrRevertMarks}{%
9495   \let\markright\@glxtr@org@markright
9496   \let\markboth\@glxtr@org@markboth
9497 }
```

\glxtrifinmark

```
9498 \newcommand*{\glxtrifinmark}[2]{#2}
```

\@glxtrinmark

```
9499 \newrobustcmd*{\@glxtrinmark}{%
9500   \let\glxtrifinmark\@firstoftwo
9501 }
```

glxtrnotinmark

```
9502 \newrobustcmd*{\@glxtrnotinmark}{%
9503   \let\glxtrifinmark\@secondoftwo
9504 }
```

\glxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```
9505 \newcommand*{\glxtrmarkhook}{%
```

Save current definitions:

```
9506 \let\@glxtr@org@MakeUppercase\MakeUppercase
9507 \let\@glxtr@org@glxtrtitleshort\glxtrtitleshort
9508 \let\@glxtr@org@glxtrtitleshortpl\glxtrtitleshortpl
9509 \let\@glxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
9510 \let\@glxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
9511 \let\@glxtr@org@glxtrtitletext\glxtrtitletext
9512 \let\@glxtr@org@Glsxtrtitletext\Glsxtrtitletext
9513 \let\@glxtr@org@glxtrtitleplural\glxtrtitleplural
9514 \let\@glxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
9515 \let\@glxtr@org@glxtrtitlefirst\glxtrtitlefirst
9516 \let\@glxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
9517 \let\@glxtr@org@glxtrtitlefirstplural\glxtrtitlefirstplural
9518 \let\@glxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
9519 \let\@glxtr@org@glxtrtitlelong\glxtrtitlelong
9520 \let\@glxtr@org@glxtrtitlelongpl\glxtrtitlelongpl
9521 \let\@glxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
9522 \let\@glxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
9523 \let\@glxtr@org@glxtrtitlefull\glxtrtitlefull
9524 \let\@glxtr@org@glxtrtitlefullpl\glxtrtitlefullpl
9525 \let\@glxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
9526 \let\@glxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl
```

New definitions

```
9527 \let\glxtrifinmark\@firstoftwo
9528 \let\MakeUppercase\MakeTextUppercase
```

```

9529 \let\glsxtrtitleshort\glsxtrheadshort
9530 \let\glsxtrtitleshortpl\glsxtrheadshortpl
9531 \let\Glsxtrtitleshort\Glsxtrheadshort
9532 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
9533 \let\glsxtrtitletext\glsxtrheadtext
9534 \let\Glsxtrtitletext\Glsxtrheadtext
9535 \let\glsxtrtitleplural\glsxtrheadplural
9536 \let\Glsxtrtitleplural\Glsxtrheadplural
9537 \let\glsxtrtitlefirst\glsxtrheadfirst
9538 \let\Glsxtrtitlefirst\Glsxtrheadfirst
9539 \let\glsxtrtitlefirstplural\glsxtrheadfirstplural
9540 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
9541 \let\glsxtrtitlelong\glsxtrheadlong
9542 \let\glsxtrtitlelongpl\glsxtrheadlongpl
9543 \let\Glsxtrtitlelong\Glsxtrheadlong
9544 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
9545 \let\glsxtrtitlefull\glsxtrheadfull
9546 \let\glsxtrtitlefullpl\glsxtrheadfullpl
9547 \let\Glsxtrtitlefull\Glsxtrheadfull
9548 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
9549 }

```

`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.)

```

9550 \newcommand*{\glsxtrrestoremarkhook}{%
9551 \let\glsxtrifinmark\@secondoftwo
9552 \let\MakeUppercase\@glsxtr@org@MakeUppercase
9553 \let\glsxtrtitleshort\@glsxtr@org@glsxtrtitleshort
9554 \let\glsxtrtitleshortpl\@glsxtr@org@glsxtrtitleshortpl
9555 \let\Glsxtrtitleshort\@glsxtr@org@Glsxtrtitleshort
9556 \let\Glsxtrtitleshortpl\@glsxtr@org@Glsxtrtitleshortpl
9557 \let\glsxtrtitletext\@glsxtr@org@glsxtrtitletext
9558 \let\Glsxtrtitletext\@glsxtr@org@Glsxtrtitletext
9559 \let\glsxtrtitleplural\@glsxtr@org@glsxtrtitleplural
9560 \let\Glsxtrtitleplural\@glsxtr@org@Glsxtrtitleplural
9561 \let\glsxtrtitlefirst\@glsxtr@org@glsxtrtitlefirst
9562 \let\Glsxtrtitlefirst\@glsxtr@org@Glsxtrtitlefirst
9563 \let\glsxtrtitlefirstplural\@glsxtr@org@glsxtrtitlefirstplural
9564 \let\Glsxtrtitlefirstplural\@glsxtr@org@Glsxtrtitlefirstplural
9565 \let\glsxtrtitlelong\@glsxtr@org@glsxtrtitlelong
9566 \let\glsxtrtitlelongpl\@glsxtr@org@glsxtrtitlelongpl
9567 \let\Glsxtrtitlelong\@glsxtr@org@Glsxtrtitlelong
9568 \let\Glsxtrtitlelongpl\@glsxtr@org@Glsxtrtitlelongpl
9569 \let\glsxtrtitlefull\@glsxtr@org@glsxtrtitlefull
9570 \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
9571 \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
9572 \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl

```

9573 }

Instead of using one document-wide conditional, use headuc attribute to determine whether or not to use the all upper case form.

`glsxtrheadshort` Command used to display short form in the page header.

```
9574 \newcommand*{\glsxtrheadshort}[1]{%
9575   \protect\NoCaseChange
9576   {%
9577     \glsifattribute{#1}{headuc}{true}%
9578     {%
9579       \GLSxtrshort [noindex,hyper=false]{#1} []%
9580     }%
9581     {%
9582       \glsxtrshort [noindex,hyper=false]{#1} []%
9583     }%
9584   }%
9585 }
```

`ltxtrtitleshort` Command to display short form of abbreviation in section title and table of contents.

```
9586 \newrobustcmd*{\ltxtrtitleshort}[1]{%
9587   \glsxtrshort [noindex,hyper=false]{#1} []%
9588 }
```

`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.

```
9589 \newcommand*{\glsxtrheadshortpl}[1]{%
9590   \protect\NoCaseChange
9591   {%
9592     \glsifattribute{#1}{headuc}{true}%
9593     {%
9594       \GLSxtrshortpl [noindex,hyper=false]{#1} []%
9595     }%
9596     {%
9597       \glsxtrshortpl [noindex,hyper=false]{#1} []%
9598     }%
9599   }%
9600 }
```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents.

```
9601 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
9602   \glsxtrshortpl [noindex,hyper=false]{#1} []%
9603 }
```

`Glsxtrheadshort` Command used to display short form in the page header with the first letter converted to upper case.

```
9604 \newcommand*{\Glsxtrheadshort}[1]{%
```

```

9605 \protect\NoCaseChange
9606 {%
9607   \glsifattribute{#1}{headuc}{true}%
9608   {%
9609     \GLSxtrshort[noindex,hyper=false]{#1} []%
9610   }%
9611   {%
9612     \GLSxtrshort[noindex,hyper=false]{#1} []%
9613   }%
9614 }%
9615 }

```

`\lsxtrtitleshort` Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

9616 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
9617   \GLSxtrshort[noindex,hyper=false]{#1} []%
9618 }

```

`\lsxtrheadshortpl` Command used to display plural short form in the page header with the first letter converted to upper case.

```

9619 \newcommand*{\Glsxtrheadshortpl}[1]{%
9620   \protect\NoCaseChange
9621   {%
9622     \glsifattribute{#1}{headuc}{true}%
9623     {%
9624       \GLSxtrshortpl[noindex,hyper=false]{#1} []%
9625     }%
9626     {%
9627       \GLSxtrshortpl[noindex,hyper=false]{#1} []%
9628     }%
9629   }%
9630 }

```

`\xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

9631 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
9632   \GLSxtrshortpl[noindex,hyper=false]{#1} []%
9633 }

```

`\glsxtrheadtext` As above but for the text value.

```

9634 \newcommand*{\glsxtrheadtext}[1]{%
9635   \protect\NoCaseChange
9636   {%
9637     \glsifattribute{#1}{headuc}{true}%
9638     {%
9639       \GLStext[noindex,hyper=false]{#1} []%
9640     }%
9641     {%

```

```

9642   \gls{text}[noindex,hyper=false]{#1}[]%
9643   }%
9644 }%
9645 }

```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```

9646 \newrobustcmd*{\glsxtrtitletext}[1]{%
9647   \gls{text}[noindex,hyper=false]{#1}[]%
9648 }

```

`\Glsxtrheadtext` First letter converted to upper case

```

9649 \newcommand*{\Glsxtrheadtext}[1]{%
9650   \protect\NoCaseChange
9651   {%
9652     \glsifattribute{#1}{headuc}{true}%
9653     {%
9654       \GLStext[noindex,hyper=false]{#1}[]%
9655     }%
9656     {%
9657       \Gls{text}[noindex,hyper=false]{#1}[]%
9658     }%
9659   }%
9660 }

```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```

9661 \newrobustcmd*{\Glsxtrtitletext}[1]{%
9662   \Gls{text}[noindex,hyper=false]{#1}[]%
9663 }

```

`lgsxtrheadplural` As above but for the plural value.

```

9664 \newcommand*{\lgsxtrheadplural}[1]{%
9665   \protect\NoCaseChange
9666   {%
9667     \glsifattribute{#1}{headuc}{true}%
9668     {%
9669       \GLSplural[noindex,hyper=false]{#1}[]%
9670     }%
9671     {%
9672       \glsplural[noindex,hyper=false]{#1}[]%
9673     }%
9674   }%
9675 }

```

`sxtrtitleplural` Command to display plural value in section title and table of contents.

```

9676 \newrobustcmd*{\sxtrtitleplural}[1]{%
9677   \glsplural[noindex,hyper=false]{#1}[]%
9678 }

```

`lsxtrheadplural` Convert first letter to upper case.

```
9679 \newcommand*{\Glsxtrheadplural}[1]{%
9680   \protect\NoCaseChange
9681   {%
9682     \glsifattribute{#1}{headuc}{true}%
9683     {%
9684       \GLSplural[noindex,hyper=false]{#1}[]%
9685     }%
9686     {%
9687       \Glsplural[noindex,hyper=false]{#1}[]%
9688     }%
9689   }%
9690 }
```

`sxtrtitleplural` Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
9691 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
9692   \Glsplural[noindex,hyper=false]{#1}[]%
9693 }
```

`glsxtrheadfirst` As above but for the first value.

```
9694 \newcommand*{\glsxtrheadfirst}[1]{%
9695   \protect\NoCaseChange
9696   {%
9697     \glsifattribute{#1}{headuc}{true}%
9698     {%
9699       \GLSfirst[noindex,hyper=false]{#1}[]%
9700     }%
9701     {%
9702       \glsfirst[noindex,hyper=false]{#1}[]%
9703     }%
9704   }%
9705 }
```

`lsxtrtitlefirst` Command to display first value in section title and table of contents.

```
9706 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
9707   \glsfirst[noindex,hyper=false]{#1}[]%
9708 }
```

`Glsxtrheadfirst` First letter converted to upper case

```
9709 \newcommand*{\Glsxtrheadfirst}[1]{%
9710   \protect\NoCaseChange
9711   {%
9712     \glsifattribute{#1}{headuc}{true}%
9713     {%
9714       \GLSfirst[noindex,hyper=false]{#1}[]%
9715     }%
9716     {%
```

```

9717   \Glsfirst[noindex,hyper=false]{#1}[]%
9718   }%
9719 }%
9720 }

```

`\lsxtrtitlefirst` Command to display first value in section title and table of contents with the first letter changed to upper case.

```

9721 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
9722   \Glsfirst[noindex,hyper=false]{#1}[]%
9723 }

```

`\headfirstplural` As above but for the firstplural value.

```

9724 \newcommand*{\Glsxtrheadfirstplural}[1]{%
9725   \protect\NoCaseChange
9726   {%
9727     \glsifattribute{#1}{headuc}{true}%
9728     {%
9729       \GLSfirstplural[noindex,hyper=false]{#1}[]%
9730     }%
9731     {%
9732       \glsfirstplural[noindex,hyper=false]{#1}[]%
9733     }%
9734   }%
9735 }

```

`\itlefirstplural` Command to display firstplural value in section title and table of contents.

```

9736 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
9737   \glsfirstplural[noindex,hyper=false]{#1}[]%
9738 }

```

`\headfirstplural` First letter converted to upper case

```

9739 \newcommand*{\Glsxtrheadfirstplural}[1]{%
9740   \protect\NoCaseChange
9741   {%
9742     \glsifattribute{#1}{headuc}{true}%
9743     {%
9744       \GLSfirstplural[noindex,hyper=false]{#1}[]%
9745     }%
9746     {%
9747       \Glsfirstplural[noindex,hyper=false]{#1}[]%
9748     }%
9749   }%
9750 }

```

`\itlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```

9751 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
9752   \Glsfirstplural[noindex,hyper=false]{#1}[]%
9753 }

```

`\glsxtrheadlong` Command used to display long form in the page header.

```
9754 \newcommand*{\glsxtrheadlong}[1]{%
9755   \protect\NoCaseChange
9756   {%
9757     \glsifattribute{#1}{headuc}{true}%
9758     {%
9759       \GLSxtrlong[noindex,hyper=false]{#1}[]%
9760     }%
9761     {%
9762       \glsxtrlong[noindex,hyper=false]{#1}[]%
9763     }%
9764   }%
9765 }
```

`glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
9766 \newrobustcmd*{\glsxtrtitlelong}[1]{%
9767   \glsxtrlong[noindex,hyper=false]{#1}[]%
9768 }
```

`lgsxtrheadlongpl` 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.

```
9769 \newcommand*{\glsxtrheadlongpl}[1]{%
9770   \protect\NoCaseChange
9771   {%
9772     \glsifattribute{#1}{headuc}{true}%
9773     {%
9774       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
9775     }%
9776     {%
9777       \glsxtrlongpl[noindex,hyper=false]{#1}[]%
9778     }%
9779   }%
9780 }
```

`lgsxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```
9781 \newrobustcmd*{\glsxtrtitlelongpl}[1]{%
9782   \glsxtrlongpl[noindex,hyper=false]{#1}[]%
9783 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```
9784 \newcommand*{\Glsxtrheadlong}[1]{%
9785   \protect\NoCaseChange
9786   {%
9787     \glsifattribute{#1}{headuc}{true}%
9788     {%
9789       \GLSxtrlong[noindex,hyper=false]{#1}[]%

```



```

9790 }%
9791 {%
9792   \Glsxtrlong[noindex,hyper=false]{#1}[]%
9793 }%
9794 }%
9795 }

```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

9796 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
9797   \Glsxtrlong[noindex,hyper=false]{#1}[]%
9798 }

```

`lgsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```

9799 \newcommand*{\Glsxtrheadlongpl}[1]{%
9800   \protect\NoCaseChange
9801   {%
9802     \glsifattribute{#1}{headuc}{true}%
9803     {%
9804       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
9805     }%
9806     {%
9807       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
9808     }%
9809   }%
9810 }

```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

9811 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
9812   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
9813 }

```

`\glsxtrheadfull` Command used to display full form in the page header.

```

9814 \newcommand*{\glsxtrheadfull}[1]{%
9815   \protect\NoCaseChange
9816   {%
9817     \glsifattribute{#1}{headuc}{true}%
9818     {%
9819       \Glsxtrfull[noindex,hyper=false]{#1}[]%
9820     }%
9821     {%
9822       \glsxtrfull[noindex,hyper=false]{#1}[]%
9823     }%
9824   }%
9825 }

```

`glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents.

```

9826 \newrobustcmd*{\glsxtrtitlefull}[1]{%
9827   \glsxtrfull[noindex,hyper=false]{#1}[]%
9828 }

```

`lgsxtrheadfullpl` 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.

```

9829 \newcommand*{\glsxtrheadfullpl}[1]{%
9830   \protect\NoCaseChange
9831   {%
9832     \glsifattribute{#1}{headuc}{true}%
9833     {%
9834       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
9835     }%
9836     {%
9837       \glsxtrfullpl[noindex,hyper=false]{#1}[]%
9838     }%
9839   }%
9840 }

```

`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents.

```

9841 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
9842   \glsxtrfullpl[noindex,hyper=false]{#1}[]%
9843 }

```

`\GLsxtrheadfull` Command used to display full form in the page header with the first letter converted to upper case.

```

9844 \newcommand*{\GLsxtrheadfull}[1]{%
9845   \protect\NoCaseChange
9846   {%
9847     \glsifattribute{#1}{headuc}{true}%
9848     {%
9849       \GLSxtrfull[noindex,hyper=false]{#1}[]%
9850     }%
9851     {%
9852       \GLsxtrfull[noindex,hyper=false]{#1}[]%
9853     }%
9854   }%
9855 }

```

`GLsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

9856 \newrobustcmd*{\GLsxtrtitlefull}[1]{%
9857   \GLsxtrfull[noindex,hyper=false]{#1}[]%
9858 }

```

`\lsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```
9859 \newcommand*{\Glsxtrheadfullpl}[1]{%
9860   \protect\NoCaseChange
9861   {%
9862     \glsifattribute{#1}{headuc}{true}%
9863     {%
9864       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
9865     }%
9866     {%
9867       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
9868     }%
9869   }%
9870 }
```

`\lsxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
9871 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
9872   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
9873 }
```

`\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.

```
9874 \ifdef\texorpdfstring
9875 {
9876   \newcommand*{\glsfmtshort}[1]{%
9877     \texorpdfstring
9878     {\glsxtrtitleshort{#1}}%
9879     {\glsentryshort{#1}}%
9880   }
9881 }
9882 {
9883   \newcommand*{\glsfmtshort}[1]{%
9884     \glsxtrtitleshort{#1}}
9885 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
9886 \ifdef\texorpdfstring
9887 {
9888   \newcommand*{\glsfmtshortpl}[1]{%
9889     \texorpdfstring
9890     {\glsxtrtitleshortpl{#1}}%
9891     {\glsentryshortpl{#1}}%
9892   }
9893 }
9894 {
9895   \newcommand*{\glsfmtshortpl}[1]{%
9896     \glsxtrtitleshortpl{#1}}
9897 }
```

```
9896 \glstrtitleshortpl{#1}}
9897 }
```

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).

```
9898 \ifdef\texorpdfstring
9899 {
9900 \newcommand*{\Glsfmtshort}[1]{%
9901 \texorpdfstring
9902 {\Glsxtrtitleshort{#1}}%
9903 {\glsentryshort{#1}}%
9904 }
9905 }
9906 {
9907 \newcommand*{\Glsfmtshort}[1]{%
9908 \Glsxtrtitleshort{#1}}
9909 }
```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```
9910 \ifdef\texorpdfstring
9911 {
9912 \newcommand*{\Glsfmtshortpl}[1]{%
9913 \texorpdfstring
9914 {\Glsxtrtitleshortpl{#1}}%
9915 {\glsentryshortpl{#1}}%
9916 }
9917 }
9918 {
9919 \newcommand*{\Glsfmtshortpl}[1]{%
9920 \Glsxtrtitleshortpl{#1}}
9921 }
```

`\glsfmttext` As above but for the text value.

```
9922 \ifdef\texorpdfstring
9923 {
9924 \newcommand*{\glsfmttext}[1]{%
9925 \texorpdfstring
9926 {\glsxtrtitletext{#1}}%
9927 {\glsentrytext{#1}}%
9928 }
9929 }
9930 {
9931 \newcommand*{\glsfmttext}[1]{%
9932 \glsxtrtitletext{#1}}
9933 }
```

`\Glsfmttext` First letter converted to upper case.

```

9934 \ifdef\texorpdfstring
9935 {
9936   \newcommand*\Glsfmttext}[1]{%
9937     \texorpdfstring
9938     {\Glsxtrtitletext{#1}}%
9939     {\glsentrytext{#1}}%
9940   }
9941 }
9942 {
9943   \newcommand*\Glsfmttext}[1]{%
9944     \Glsxtrtitletext{#1}}
9945 }

```

`\glsfmtplural` As above but for the plural value.

```

9946 \ifdef\texorpdfstring
9947 {
9948   \newcommand*\glsfmtplural}[1]{%
9949     \texorpdfstring
9950     {\glsxtrtitleplural{#1}}%
9951     {\glsentryplural{#1}}%
9952   }
9953 }
9954 {
9955   \newcommand*\glsfmtplural}[1]{%
9956     \glsxtrtitleplural{#1}}
9957 }

```

`\Glsfmtplural` First letter converted to upper case.

```

9958 \ifdef\texorpdfstring
9959 {
9960   \newcommand*\Glsfmtplural}[1]{%
9961     \texorpdfstring
9962     {\Glsxtrtitleplural{#1}}%
9963     {\glsentryplural{#1}}%
9964   }
9965 }
9966 {
9967   \newcommand*\Glsfmtplural}[1]{%
9968     \Glsxtrtitleplural{#1}}
9969 }

```

`\glsfmtfirst` As above but for the first value.

```

9970 \ifdef\texorpdfstring
9971 {
9972   \newcommand*\glsfmtfirst}[1]{%
9973     \texorpdfstring
9974     {\glsxtrtitlefirst{#1}}%
9975     {\glsentryfirst{#1}}%
9976   }

```

```

9977 }
9978 {
9979 \newcommand*{\glsfmtfirst}[1]{%
9980 \glsxtrtitlefirst{#1}}
9981 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

9982 \ifdef\texorpdfstring
9983 {
9984 \newcommand*{\Glsfmtfirst}[1]{%
9985 \texorpdfstring
9986 {\Glsxtrtitlefirst{#1}}%
9987 {\glsentryfirst{#1}}%
9988 }
9989 }
9990 {
9991 \newcommand*{\Glsfmtfirst}[1]{%
9992 \Glsxtrtitlefirst{#1}}
9993 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

9994 \ifdef\texorpdfstring
9995 {
9996 \newcommand*{\glsfmtfirstpl}[1]{%
9997 \texorpdfstring
9998 {\glsxtrtitlefirstplural{#1}}%
9999 {\glsentryfirstplural{#1}}%
10000 }
10001 }
10002 {
10003 \newcommand*{\glsfmtfirstpl}[1]{%
10004 \glsxtrtitlefirstplural{#1}}
10005 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

10006 \ifdef\texorpdfstring
10007 {
10008 \newcommand*{\Glsfmtfirstpl}[1]{%
10009 \texorpdfstring
10010 {\Glsxtrtitlefirstplural{#1}}%
10011 {\glsentryfirstplural{#1}}%
10012 }
10013 }
10014 {
10015 \newcommand*{\Glsfmtfirstpl}[1]{%
10016 \Glsxtrtitlefirstplural{#1}}
10017 }

```

`\glsfmtlong` As above but for the long value.

```

10018 \ifdef\texorpdfstring
10019 {
10020   \newcommand*\glsfmtlong}[1]{%
10021     \texorpdfstring
10022     {\glsxtrtitlelong{#1}}%
10023     {\glsentrylong{#1}}%
10024   }
10025 }
10026 {
10027   \newcommand*\glsfmtlong}[1]{%
10028     \glsxtrtitlelong{#1}}
10029 }

```

`\Glsfmtlong` First letter converted to upper case.

```

10030 \ifdef\texorpdfstring
10031 {
10032   \newcommand*\Glsfmtlong}[1]{%
10033     \texorpdfstring
10034     {\Glsxtrtitlelong{#1}}%
10035     {\glsentrylong{#1}}%
10036   }
10037 }
10038 {
10039   \newcommand*\Glsfmtlong}[1]{%
10040     \Glsxtrtitlelong{#1}}
10041 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

10042 \ifdef\texorpdfstring
10043 {
10044   \newcommand*\glsfmtlongpl}[1]{%
10045     \texorpdfstring
10046     {\glsxtrtitlelongpl{#1}}%
10047     {\glsentrylongpl{#1}}%
10048   }
10049 }
10050 {
10051   \newcommand*\glsfmtlongpl}[1]{%
10052     \glsxtrtitlelongpl{#1}}
10053 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

10054 \ifdef\texorpdfstring
10055 {
10056   \newcommand*\Glsfmtlongpl}[1]{%
10057     \texorpdfstring
10058     {\Glsxtrtitlelongpl{#1}}%
10059     {\glsentrylongpl{#1}}%
10060   }

```

```

10061 }
10062 {
10063   \newcommand*{\Glsfmtlongpl}[1]{%
10064     \Glsxtrtitlelongpl{#1}}
10065 }

```

`\glsfmtfull` In-line full format.

```

10066 \ifdef\texorpdfstring
10067 {
10068   \newcommand*{\glsfmtfull}[1]{%
10069     \texorpdfstring
10070     {\glsxtrtitlefull{#1}}%
10071     {\glsxtrinlinefullformat{#1}{}}%
10072   }
10073 }
10074 {
10075   \newcommand*{\glsfmtfull}[1]{%
10076     \glsxtrtitlefull{#1}}
10077 }

```

`\Glsfmtfull` First letter converted to upper case.

```

10078 \ifdef\texorpdfstring
10079 {
10080   \newcommand*{\Glsfmtfull}[1]{%
10081     \texorpdfstring
10082     {\Glsxtrtitlefull{#1}}%
10083     {\Glsxtrinlinefullformat{#1}{}}%
10084   }
10085 }
10086 {
10087   \newcommand*{\Glsfmtfull}[1]{%
10088     \Glsxtrtitlefull{#1}}
10089 }

```

`\glsfmtfullpl` In-line full plural format.

```

10090 \ifdef\texorpdfstring
10091 {
10092   \newcommand*{\glsfmtfullpl}[1]{%
10093     \texorpdfstring
10094     {\glsxtrtitlefullpl{#1}}%
10095     {\glsxtrinlinefullplformat{#1}{}}%
10096   }
10097 }
10098 {
10099   \newcommand*{\glsfmtfullpl}[1]{%
10100     \glsxtrtitlefullpl{#1}}
10101 }

```

`\Glsfmtfullpl` First letter converted to upper case.


```

10102 \ifdef\teorpdfstring
10103 {
10104   \newcommand*\Glsfmtfullpl}[1]{%
10105     \teorpdfstring
10106     {\Glsxtrtitlefullpl{#1}}%
10107     {\Glsxtrinlinelinefullplformat{#1}{}}%
10108   }
10109 }
10110 {
10111   \newcommand*\Glsfmtfullpl}[1]{%
10112     \Glsxtrtitlefullpl{#1}}
10113 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

sariesExtraLang

```

10114 \newcommand*\RequireGlossariesExtraLang}[1]{%
10115   \ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
10116 }

```

sariesExtraLang

```

10117 \newcommand*\ProvidesGlossariesExtraLang}[1]{%
10118   \ProvidesFile{glossariesxtr-#1.ldf}%
10119 }

```

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.)

```

10120 \@ifpackageloaded{tracklang}
10121 {%
10122   \AnyTrackedLanguages
10123   {%
10124     \ForEachTrackedDialect{\this@dialect}{%
10125       \IfTrackedLanguageFileExists{\this@dialect}%
10126       {glossariesxtr-}% prefix
10127       {.ldf}%
10128       {%
10129         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
10130       }%
10131     }%
10132   }%
10133 }%
10134 }%
10135 {}%
10136 }

```

10137 {}

Load glossaries-extra-stylemods if required.

10138 \@glsxtr@redefstyles

and set the style:

10139 \@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:

```
10140 \NeedsTeXFormat{LaTeX2e}
10141 \ProvidesPackage{glossaries-extra-stylemods}[2017/09/11 v1.20 (NLCT)]
```

Provide package options to automatically load required predefined styles. The simplest method is to just test for the existence of the file `glossary-<option>.sty`. Packages can't be loaded whilst the options are being processed, so save the list in `\@glsxtr@loadstyles`.

```
sxtr@loadstyles
```

```
10142 \newcommand*{\@glsxtr@loadstyles}{%
10143 \DeclareOption*{%
10144   \IfFileExists{glossary-\CurrentOption.sty}
10145   {\eappto\@glsxtr@loadstyles{%
10146     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
10147   {\PackageError{glossaries-extra-styles}%
10148     {Unknown option '\CurrentOption'}{}}
10149 }
```

Process the package options:

```
10150 \ProcessOptions
```

Load the required packages:

```
10151 \@glsxtr@loadstyles
```

Adjust the styles that the post description hook added, but only for styles that have already been defined. All the tree styles in `glossary-tree` include the post description hook, so they don't require adjustment. Similarly for `glossary-mcols` which builds on the tree styles.

In case we have an old version of `glossaries`:

```
ewglossarystyle
```

```
10152 \providecommand{\renewglossarystyle}[2]{%
10153   \ifcsundef{@glsstyle@#1}%
10154   {%
10155     \PackageError{glossaries-extra}{Glossary style '#1' isn't already defined}{}}%
```

```

10156 }%
10157 {%
10158   \csdef{@glsstyle@#1}{#2}%
10159 }%
10160 }

```

2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying.

```

10161 \ifdefined{@glsstyle@listdotted}
10162 {%
10163   \renewglossarystyle{listdotted}{%
10164     \setglossarystyle{list}%
10165     \renewcommand*{\glossentry}[2]{%
10166       \item[]\makebox[\glslistdottedwidth][l]{%
10167         \glstryitem{##1}%
10168         \glstarget{##1}{\glossentryname{##1}}%
10169         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
10170         \glossentrydesc{##1}\glspostdescription}%
10171     \renewcommand*{\subglossentry}[3]{%
10172       \item[]\makebox[\glslistdottedwidth][l]{%
10173         \glssubentryitem{##2}%
10174         \glstarget{##2}{\glossentryname{##2}}%
10175         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
10176         \glossentrydesc{##2}\glspostdescription}%
10177   }
10178 }
10179 {}

```

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.

```

10180 \ifcsdef{@glsstyle@long3col}
10181 {%
10182   \renewglossarystyle{long3col}{%
10183     \renewenvironment{theglossary}%
10184       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
10185       {\end{longtable}}%
10186     \renewcommand*{\glossaryheader}{}%
10187     \renewcommand*{\glsgroupheading}[1]{}%
10188     \renewcommand{\glossentry}[2]{%
10189       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10190       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline

```

```

10191 }%
10192 \renewcommand{\subglossentry}[3]{%
10193     &
10194     \glssubentryitem{##2}%
10195     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10196     ##3\tabularnewline
10197 }%
10198 \renewcommand*\{glsgroupskip}{%
10199     \ifglsnogroupskip\else & &\tabularnewline\fi}%
10200 }
10201 }
10202 {}

```

Four column style:

```

10203 \ifcsdef{@glsstyle@long4col}
10204 {%
10205     \renewglossarystyle{long4col}{%
10206         \renewenvironment{theglossary}%
10207             {\begin{longtable}{llll}}%
10208             {\end{longtable}}%
10209         \renewcommand*\{glossaryheader}{}%
10210         \renewcommand*\{glsgroupheading}[1]{}%
10211         \renewcommand{\glossentry}[2]{%
10212             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10213             \glossentrydesc{##1}\glspostdescription &
10214             \glossentrysymbol{##1} &
10215             ##2\tabularnewline
10216         }%
10217         \renewcommand{\subglossentry}[3]{%
10218             &
10219             \glssubentryitem{##2}%
10220             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10221             \glossentrysymbol{##2} & ##3\tabularnewline
10222         }%
10223         \renewcommand*\{glsgroupskip}{%
10224             \ifglsnogroupskip\else & &\tabularnewline\fi}%
10225     }
10226 }
10227 {}

```

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.

```

10228 \ifcsdef{@glsstyle@longragged3col}
10229 {%
10230     \renewglossarystyle{longragged3col}{%

```

```

10231 \renewenvironment{theglossary}%
10232   {\begin{longtable}[l>{\raggedright}p{\glsdescwidth}%
10233     >{\raggedright}p{\glspagelistwidth}}}%
10234   {\end{longtable}}}%
10235 \renewcommand*{\glossaryheader}{}%
10236 \renewcommand*{\glsgroupheading}[1]{}%
10237 \renewcommand{\glossentry}[2]{%
10238   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10239   \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
10240 }%
10241 \renewcommand{\subglossentry}[3]{%
10242   &
10243   \glssubentryitem{##2}%
10244   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10245   ##3\tabularnewline
10246 }%
10247 \renewcommand*{\glsgroupskip}{%
10248   \ifglsnogroupskip\else & &\tabularnewline\fi}%
10249 }
10250 }
10251 {}

```

Four column style:

```

10252 \ifcsdef{@glsstyle@altlongragged4col}
10253 {%
10254   \renewglossarystyle{altlongragged4col}{%
10255     \renewenvironment{theglossary}%
10256       {\begin{longtable}[l>{\raggedright}p{\glsdescwidth}l%
10257         >{\raggedright}p{\glspagelistwidth}}}%
10258       {\end{longtable}}}%
10259     \renewcommand*{\glossaryheader}{}%
10260     \renewcommand*{\glsgroupheading}[1]{}%
10261     \renewcommand{\glossentry}[2]{%
10262       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10263       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
10264       ##2\tabularnewline
10265     }%
10266     \renewcommand{\subglossentry}[3]{%
10267       &
10268       \glssubentryitem{##2}%
10269       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10270       \glossentrysymbol{##2} & ##3\tabularnewline
10271     }%
10272     \renewcommand*{\glsgroupskip}{%
10273       \ifglsnogroupskip\else & & &\tabularnewline\fi}%
10274   }
10275 }
10276 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment, but not the two column styles.

```
10277 \ifcsdef{@glsstyle@super3col}
10278 {%
10279   \renewglossarystyle{super3col}{%
10280     \renewenvironment{theglossary}%
10281       {\tablehead{}}\tabletail{}}%
10282     \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}%
10283     {\end{supertabular}}%
10284     \renewcommand*{\glossaryheader}{}%
10285     \renewcommand*{\glsgroupheading}[1]{}%
10286     \renewcommand{\glossentry}[2]{%
10287       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10288       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
10289     }%
10290     \renewcommand{\subglossentry}[3]{%
10291       &
10292       \glssubentryitem{##2}%
10293       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10294       ##3\tabularnewline
10295     }%
10296     \renewcommand*{\glsgroupskip}{%
10297       \ifglsnogroupskip\else & \tabularnewline\fi}%
10298   }
10299 }
10300 {}
```

Four column styles:

```
10301 \ifcsdef{@glsstyle@super4col}
10302 {%
10303   \renewglossarystyle{super4col}{%
10304     \renewenvironment{theglossary}%
10305       {\tablehead{}}\tabletail{}}%
10306     \begin{supertabular}{lllll}%
10307     \end{supertabular}}%
10308     \renewcommand*{\glossaryheader}{}%
10309     \renewcommand*{\glsgroupheading}[1]{}%
10310     \renewcommand{\glossentry}[2]{%
10311       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10312       \glossentrydesc{##1}\glspostdescription &
10313       \glossentrysymbol{##1} & ##2\tabularnewline
10314     }%
10315     \renewcommand{\subglossentry}[3]{%
10316       &
10317       \glssubentryitem{##2}%
10318       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10319       \glossentrysymbol{##2} & ##3\tabularnewline
10320     }%
10321     \renewcommand*{\glsgroupskip}{%

```

```

10322     \ifglsnogroupskip\else & & \tabularnewline\fi}%
10323   }
10324 }
10325 {}

```

2.6 Super Ragged Styles

The three and four column styles require adjustment, but not the two column styles.

```

10326 \ifcsdef{@glsstyle@superragged3col}
10327 {%
10328   \renewglossarystyle{superragged3col}{%
10329     \renewenvironment{theglossary}%
10330       {\tablehead{ }\tabletail{ }}%
10331       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%
10332         >{\raggedright}p{\glspagelistwidth}}}%
10333       {\end{supertabular}}}%
10334   \renewcommand*{\glossaryheader}{ }%
10335   \renewcommand*{\glsgroupheading}[1]{ }%
10336   \renewcommand{\glossentry}[2]{%
10337     \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10338     \glossentrydesc{##1}\glspostdescription &
10339     ##2\tabularnewline
10340   }%
10341   \renewcommand{\subglossentry}[3]{%
10342     &
10343     \glssubentryitem{##2}%
10344     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10345     ##3\tabularnewline
10346   }%
10347   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
10348     &\tabularnewline\fi}%
10349 }
10350 }
10351 {}

```

Four columns:

```

10352 \ifcsdef{@glsstyle@altsuperragged4col}
10353 {%
10354   \renewglossarystyle{altsuperragged4col}{%
10355     \renewenvironment{theglossary}%
10356       {\tablehead{ }\tabletail{ }}%
10357       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}1%
10358         >{\raggedright}p{\glspagelistwidth}}}%
10359       {\end{supertabular}}}%
10360   \renewcommand*{\glossaryheader}{ }%
10361   \renewcommand{\glossentry}[2]{%
10362     \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
10363     \glossentrydesc{##1}\glspostdescription &
10364     \glossentrysymbol{##1} & ##2\tabularnewline

```



```

10365 }%
10366 \renewcommand{\subglossentry}[3]{%
10367     &
10368     \glssubentryitem{##2}%
10369     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
10370     \glossentrysymbol{##2} & ##3\tabularnewline
10371 }%
10372 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & &
10373     &\tabularnewline\fi}%
10374 }
10375 }
10376 {}

```

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`.

```

10377 \ifdef{\@glsstyle@inline}
10378 {%
10379     \renewcommand*{\glspostinline}{.\spacefactor\sfcode‘\.’}

```

Just use `\glsxtrpostdescription` instead of `\glspostdescription`.

```

10380     \renewcommand*{\glsinlinedescformat}[3]{%
10381         \space#1\glsxtrpostdescription}
10382     \renewcommand*{\glsinlinesubdescformat}[3]{%
10383         #1\glsxtrpostdescription}
10384 }
10385 {}

```

2.8 Tree Styles

The `almtree` style is redefined to make it easier to made minor adjustments.

```

10386 \ifdef{\@glsstyle@almtree}
10387 {%

```

Only redefine this style if it's already been defined.

SymbolDescLocation

```
\glxtralmtreeSymbolDescLocation{\langle label \rangle}{\langle location list \rangle}
```

Layout the symbol, description and location for top-level entries.

```

10388 \newcommand{\glxtralmtreeSymbolDescLocation}[2]{%
10389     {%
10390         \let\par\glxtrAltTreePar

```

```

10391     \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
10392     \glossentrydesc{#1}\glspostdescription \space #2\par
10393     }%
10394     }

```

`trAltTreeIndent` Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```
10395 \newlength\glxtrAltTreeIndent
```

`lsxtrAltTreePar` Multi-paragraph descriptions need to keep the hanging indent.

```

10396 \newcommand{\glxtrAltTreePar}{%
10397     \@@par
10398     \glxtrAltTreeSetHangIndent
10399     \setlength{\parindent}{\dimexpr\hangindent+\glxtrAltTreeIndent}%
10400     }

```

`symbolDescLocation` `\glxtralmtreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

10401 \newcommand{\glxtralmtreeSubSymbolDescLocation}[3]{%
10402     \glxtralmtreeSymbolDescLocation{#2}{#3}%
10403     }

```

`trreetopindent` 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.

```
10404 \newlength\glxtrtreetopindent
```

`lsxtralmtreeInit` User-level initialisation for the almtree style.

```

10405 \newcommand*{\lsxtralmtreeInit}{%
10406     \settowidth{\glxtrtreetopindent}{\glstreenamfmt{\glsgetwidestname\space}}%
10407     \glxtrAltTreeIndent=\parindent
10408     }

```

`\eglissetwidest` The original `\glissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

10409 \newcommand*{\eglissetwidest}[2][0]{%
10410     \protected@csedef{@glswidestname\romannumeral#1}{#2}%
10411     }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```

10412 \newcommand*{\xglissetwidest}[2][0]{%
10413     \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
10414     }

```

`lsgsetwidestname` Provide a user-level macro to obtain the widest top-level name.

```
10415 \newcommand*{\glsggetwidestname}{\@glswidestname}
```

`etwidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```
10416 \newcommand*\glsgetwidestsubname}[1]{%
10417   \ifcsundef{@glswidestname\romannumeral#1}%
10418     {\@glswidestname}%
10419     {\csuse{@glswidestname\romannumeral#1}}%
10420 }
```

`estTopLevelName` CamelCase is easier for long command names. Provide a CamelCase synonym of `\glsfindwidesttoplevelname`

```
10421 \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.

```
10422 \newrobustcmd*\glsFindWidestUsedTopLevelName}[1][\@glo@types]{%
10423   \dimen@=0pt\relax
10424   \gls@tmplen=0pt\relax
10425   \forallglossaries[#1]{\@gls@type}%
10426   {%
10427     \forglentries[\@gls@type]{\@glo@label}%
10428     {%
10429       \ifglsused{\@glo@label}%
10430       {%
10431         \ifglshasparent{\@glo@label}%
10432         {}%
10433         {%
10434           \settowidth{\dimen@}%
10435             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
10436           \ifdim\dimen@>\gls@tmplen
10437             \gls@tmplen=\dimen@
10438           \eglssetwidest{\glsentryname{\@glo@label}}%
10439           \fi
10440         }%
10441       }%
10442     }%
10443   }%
10444 }%
10445 }
```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```
10446 \newrobustcmd*\glsFindWidestUsedAnyName}[1][\@glo@types]{%
10447   \dimen@=0pt\relax
10448   \gls@tmplen=0pt\relax
10449   \forallglossaries[#1]{\@gls@type}%
10450   {%
10451     \forglentries[\@gls@type]{\@glo@label}%
10452     {%
```

```

10453     \ifglsused{\@glo@label}%
10454     {%
10455         \settowidth{\dimen@}%
10456         {\glstreenamefmt{\glstentryname{\@glo@label}}}%
10457         \ifdim\dimen@>\gls@tmplen
10458             \gls@tmplen=\dimen@
10459             \eglssetwidest{\glstentryname{\@glo@label}}%
10460         \fi
10461     }%
10462     {%
10463     }%
10464 }%
10465 }

```

`FindWidestAnyName` Like the above but doesn't check if the entry has been used.

```

10466 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
10467     \dimen@=0pt\relax
10468     \gls@tmplen=0pt\relax
10469     \forallglossaries[#1]{\@gls@type}%
10470     {%
10471         \forglentries[\@gls@type]{\@glo@label}%
10472         {%
10473             \settowidth{\dimen@}%
10474             {\glstreenamefmt{\glstentryname{\@glo@label}}}%
10475             \ifdim\dimen@>\gls@tmplen
10476                 \gls@tmplen=\dimen@
10477                 \eglssetwidest{\glstentryname{\@glo@label}}%
10478             \fi
10479         }%
10480     }%
10481 }

```

`FindWidestUsedLevelTwo` This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well.

Any entry that has a great-grandparent is ignored.

```

10482 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
10483     \dimen@=0pt\relax
10484     \dimen@i=0pt\relax
10485     \dimen@ii=0pt\relax
10486     \forallglossaries[#1]{\@gls@type}%
10487     {%
10488         \forglentries[\@gls@type]{\@glo@label}%
10489         {%
10490             \ifglsused{\@glo@label}%
10491             {%
10492                 \ifglshasparent{\@glo@label}%
10493                 {%
10494                     \edef\@glo@parent{\csuse{glo@glstetoklabel}{\@glo@label}@parent}}%
10495                     \ifglshasparent{\@glo@parent}%
10496                 }%

```

```

10497         \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
10498         \ifglshasparent{\@glo@parent}%
10499         {%
10500         {%
10501             \settowidth{\gls@tmplen}%
10502             {\glsstreenamfmt{\glsentryname{\@glo@label}}}%
10503             \ifdim\gls@tmplen>\dimen@ii
10504             \dimen@ii=\gls@tmplen
10505             \eglssetwidest[2]{\glsentryname{\@glo@label}}%
10506             \fi
10507         }%
10508     }%
10509     {%
10510         \settowidth{\gls@tmplen}%
10511         {\glsstreenamfmt{\glsentryname{\@glo@label}}}%
10512         \ifdim\gls@tmplen>\dimen@i
10513         \dimen@i=\gls@tmplen
10514         \eglssetwidest[1]{\glsentryname{\@glo@label}}%
10515         \fi
10516     }%
10517 }%
10518 {%
10519     \settowidth{\gls@tmplen}%
10520     {\glsstreenamfmt{\glsentryname{\@glo@label}}}%
10521     \ifdim\gls@tmplen>\dimen@
10522     \dimen@=\gls@tmplen
10523     \eglssetwidest{\glsentryname{\@glo@label}}%
10524     \fi
10525 }%
10526 }%
10527 }%
10528 }%
10529 }%
10530 }

```

`\widestLevelTwo` This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

10531 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
10532     \dimen@=0pt\relax
10533     \dimen@i=0pt\relax
10534     \dimen@ii=0pt\relax
10535     \forallglossaries[#1]{\@gls@type}%
10536     {%
10537         \forglseries[\@gls@type]{\@glo@label}%
10538         {%
10539             \ifglshasparent{\@glo@label}%
10540             {%
10541                 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
10542                 \ifglshasparent{\@glo@parent}%
10543                 {%

```

```

10544     \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
10545     \ifglshasparent{\@glo@parent}%
10546     {}%
10547     {%
10548         \settowidth{\gls@tmplen}%
10549             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10550     \ifdim\gls@tmplen>\dimen@ii
10551         \dimen@ii=\gls@tmplen
10552         \eglssetwidest[2]{\glsentryname{\@glo@label}}%
10553     \fi
10554     }%
10555 }%
10556 {%
10557     \settowidth{\gls@tmplen}%
10558         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10559 \ifdim\gls@tmplen>\dimen@i
10560     \dimen@i=\gls@tmplen
10561     \eglssetwidest[1]{\glsentryname{\@glo@label}}%
10562 \fi
10563 }%
10564 }%
10565 {%
10566     \settowidth{\gls@tmplen}%
10567         {\glsentryname{\@glo@label}}%
10568 \ifdim\gls@tmplen>\dimen@
10569     \dimen@=\gls@tmplen
10570     \eglssetwidest{\glsentryname{\@glo@label}}%
10571 \fi
10572 }%
10573 }%
10574 }%
10575 }

```

`\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.

```

10576 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
10577     \dimen@=0pt\relax
10578     \gls@tmplen=0pt\relax
10579     #2=0pt\relax
10580     \forallglossaries[#1]{\@gls@type}%
10581     {%
10582         \forglsentries[\@gls@type]{\@glo@label}%
10583         {%
10584             \ifglssused{\@glo@label}%
10585             {%
10586                 \settowidth{\dimen@}%
10587                     {\glstreenamefmt{\glsentryname{\@glo@label}}}%
10588                 \ifdim\dimen@>\gls@tmplen
10589                     \gls@tmplen=\dimen@

```

```

10590         \eglssetwidest{\glsentryname{\@glo@label}}%
10591         \fi
10592         \settowidth{\dimen@}%
10593         {\glsentrysymbol{\@glo@label}}%
10594         \ifdim\dimen@>#2\relax
10595             #2=\dimen@
10596         \fi
10597     }%
10598     {}%
10599 }%
10600 }%
10601 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

10602 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
10603     \dimen@=0pt\relax
10604     \gls@tmplen=0pt\relax
10605     #2=0pt\relax
10606     \forallglossaries[#1]{\@gls@type}%
10607     {%
10608         \forglsentries[\@gls@type]{\@glo@label}%
10609         {%
10610             \settowidth{\dimen@}%
10611             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
10612             \ifdim\dimen@>\gls@tmplen
10613                 \gls@tmplen=\dimen@
10614                 \eglssetwidest{\glsentryname{\@glo@label}}%
10615             \fi
10616             \settowidth{\dimen@}%
10617             {\glsentrysymbol{\@glo@label}}%
10618             \ifdim\dimen@>#2\relax
10619                 #2=\dimen@
10620             \fi
10621         }%
10622     }%
10623 }

```

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.

```

10624 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
10625     \dimen@=0pt\relax
10626     \gls@tmplen=0pt\relax
10627     #2=0pt\relax
10628     #3=0pt\relax
10629     \forallglossaries[#1]{\@gls@type}%
10630     {%
10631         \forglsentries[\@gls@type]{\@glo@label}%

```

```

10632   {%
10633     \ifglsused{\@glo@label}%
10634     {%
10635       \settowidth{\dimen@}%
10636       {\glstreenamfmt{\glstentryname{\@glo@label}}}%
10637       \ifdim\dimen@>\gls@tmplen
10638         \gls@tmplen=\dimen@
10639         \eglssetwidest{\glstentryname{\@glo@label}}%
10640       \fi
10641       \settowidth{\dimen@}%
10642       {\glstentrysymbol{\@glo@label}}%
10643       \ifdim\dimen@>#2\relax
10644         #2=\dimen@
10645       \fi
10646       \settowidth{\dimen@}%
10647       {\GlsXtrFormatLocationList{\glstentrynumberlist{\@glo@label}}}%
10648       \ifdim\dimen@>#3\relax
10649         #3=\dimen@
10650       \fi
10651     }%
10652   }%
10653 }%
10654 }%
10655 }

```

`\eSymbolLocation` Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

10656 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
10657   \dimen@=0pt\relax
10658   \gls@tmplen=0pt\relax
10659   #2=0pt\relax
10660   #3=0pt\relax
10661   \forallglossaries[#1]{\@gls@type}%
10662   {%
10663     \forglentries[\@gls@type]{\@glo@label}%
10664     {%
10665       \settowidth{\dimen@}%
10666       {\glstreenamfmt{\glstentryname{\@glo@label}}}%
10667       \ifdim\dimen@>\gls@tmplen
10668         \gls@tmplen=\dimen@
10669         \eglssetwidest{\glstentryname{\@glo@label}}%
10670       \fi
10671       \settowidth{\dimen@}%
10672       {\glstentrysymbol{\@glo@label}}%
10673       \ifdim\dimen@>#2\relax
10674         #2=\dimen@
10675       \fi
10676       \settowidth{\dimen@}%
10677       {\GlsXtrFormatLocationList{\glstentrynumberlist{\@glo@label}}}%
10678     \ifdim\dimen@>#3\relax

```



```

10679         #3=\dimen@
10680         \fi
10681     }%
10682 }%
10683 }

```

`\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.

```

10684 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
10685     \dimen@=0pt\relax
10686     \gls@tmplen=0pt\relax
10687     #2=0pt\relax
10688     \forallglossaries[#1]{\@gls@type}%
10689     {%
10690         \forglstentries[\@gls@type]{\@glo@label}%
10691         {%
10692             \ifglsused{\@glo@label}%
10693             {%
10694                 \settowidth{\dimen@}%
10695                 {\glstreenamefmt{\glstentryname{\@glo@label}}}%
10696                 \ifdim\dimen@>\gls@tmplen
10697                     \gls@tmplen=\dimen@
10698                     \eglssetwidest{\glstentryname{\@glo@label}}%
10699                 \fi
10700                 \settowidth{\dimen@}%
10701                 {\GlsXtrFormatLocationList{\glstentrynumberlist{\@glo@label}}}%
10702                 \ifdim\dimen@>#2\relax
10703                     #2=\dimen@
10704                 \fi
10705             }%
10706         }%
10707     }%
10708 }%
10709 }

```

`\AnyNameLocation` Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

10710 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
10711     \dimen@=0pt\relax
10712     \gls@tmplen=0pt\relax
10713     #2=0pt\relax
10714     \forallglossaries[#1]{\@gls@type}%
10715     {%
10716         \forglstentries[\@gls@type]{\@glo@label}%
10717         {%
10718             \settowidth{\dimen@}%
10719             {\glstreenamefmt{\glstentryname{\@glo@label}}}%
10720             \ifdim\dimen@>\gls@tmplen
10721                 \gls@tmplen=\dimen@

```

```

10722     \eglssetwidest{\glstryname{\@glo@label}}%
10723     \fi
10724     \settowidth{\dimen@}%
10725     {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
10726     \ifdim\dimen@>#2\relax
10727         #2=\dimen@
10728     \fi
10729 }%
10730 }%
10731 }

```

`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`.

```

10732 \newcommand*\glsxtrComputeTreeIndent}[1]{%
10733     \glstreeindent=\glsxtrtreetopindent\relax
10734 }

```

`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.

```

10735 \newcommand*\glsxtrComputeTreeSubIndent}[3]{%
10736     \ifcsundef{@glswidestname\romannumeral#1}%
10737     {%
10738         \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
10739     }%
10740     {%
10741         \settowidth{#3}{\glstreenamefmt{%
10742             \csname @glswidestname\romannumeral#1\endcsname\space}}%
10743     }%
10744 }

```

`treeSetHangIndent` Set `\hangindent` for top-level entries:

```

10745 \newcommand*\glsxtrAltTreeSetHangIndent{\hangindent\glstreeindent}

```

`treeSetSubHangIndent` Set `\hangindent` for sub-entries:

```

10746 \newcommand*\glsxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine `almtree`:

```

10747 \renewglossarystyle{almtree}{%
10748     \renewenvironment{theglossary}%
10749     {%
10750     \glsxtralmtreeInit

```

```

10751     \def\@gls@prevlevel{-1}%
10752     \mbox{}\par}%
10753     {\par}%
10754 \renewcommand*\glossaryheader{}%
10755 \renewcommand*\glsgroupheading}[1]{}%
10756 \renewcommand\glossentry}[2]{%
10757     \ifnum\@gls@prevlevel=0\relax
10758     \else
10759         \glxtrComputeTreeIndent{##1}%
10760     \fi
10761     \parindent\glstreeindent
10762     \glxtrAltTreeSetHangIndent
10763     \makebox[Opt][r]%
10764     {%
10765         \glstreenamebox{\glstreeindent}%
10766         {%
10767             \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
10768             \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
10769         }%
10770     }%
10771     \glxtralttreeSymbolDescLocation{##1}{##2}%
10772     \def\@gls@prevlevel{0}%
10773 }
10774 \renewcommand\subglossentry}[3]{%
10775     \ifnum##1=1\relax
10776         \glssubentryitem{##2}%
10777     \fi
10778     \ifnum\@gls@prevlevel=##1\relax
10779     \else
10780         \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
10781         \ifnum\@gls@prevlevel<##1\relax
10782             \setlength\glstreeindent\gls@tmplen
10783             \addtolength\glstreeindent\parindent
10784             \parindent\glstreeindent
10785         \else
10786             \ifnum\@gls@prevlevel=0\relax
10787                 \glxtrComputeTreeIndent{##2}%
10788             \else
10789                 \glxtrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
10790             \fi
10791             \addtolength\parindent{-\glstreeindent}%
10792             \setlength\glstreeindent\parindent
10793         \fi
10794     \fi
10795     \glxtrAltTreeSetSubHangIndent{##1}%
10796     \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
10797         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
10798     \glxtralttreeSubSymbolDescLocation{##1}{##2}{##3}%
10799     \def\@gls@prevlevel{##1}%

```

```
10800 }%
10801 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
10802 }
10803 }%
10804 {%
```

Assume the style isn't required if it hasn't already been defined.

```
10805 }
```

Reset the default style

```
10806 \ifx\@glossary@default@style\relax
10807 \else
10808 \setglossarystyle{\@glsxtr@current@style}
10809 \fi
```

Glossary

First use The first time a glossary entry is used (from the start of the document or after a reset) with one of the following commands: `\gls`, `\Gls`, `\GLS`, `\glspl`, `\Glspl`, `\GLSpl` or `\glsdisp`. *see* **first use flag** & **first use text**

First use flag A conditional that determines whether or not the entry has been used according to the rules of **first use**.

First use text The text that is displayed on **first use**, which is governed by the first and first-plural keys of `\newglossaryentry`. (May be overridden by `\glsdisp`.)

`makeindex` An indexing application.

`xindy` An flexible indexing application with multilingual support written in Perl.

Change History

0.1 (2015-11-22)	
General: Initial experimental release	5
0.2 (2015-11-30)	
\Glsfmtshort: new	276
\glsfmtshort: new	275
\Glsfmtshortpl: new	276
\glsfmtshortpl: new	275
short: switched inline full form to short (long)	184
0.3 (2015-12-02)	
\@ACRlong: added redefinition	63
\@ACRlongpl: added redefinition	64
\@ACRshort: added redefinition	61
\@ACRshortpl: added redefinition	62
\@Acrlong: added redefinition	62
\@Acrlongpl: added redefinition	63
\@Acrshort: added redefinition	60
\@Acrshortpl: added redefinition	61
\@GLSdesc: added redefinition	56
\@GLSdescplural@: added redefinition	57
\@GLSfirst@: added redefinition	54
\@GLSfirstplural@: added redefinition	55
\@GLSname@: added redefinition	56
\@GLSplural@: added redefinition	55
\@GLSsymbol@: added redefinition	57
\@GLSsymbolplural@: added redefinition	58
\@GLStext@: added redefinition	53
\@GLSuseri@: added redefinition	58
\@GLSuserii@: added redefinition	59
\@GLSuseriii@: added redefinition	59
\@GLSuseriv@: added redefinition	59
\@GLSuseriv@: added redefinition	60
\@GLSuseriv@: added redefinition	60
\@Glsdesc@: added redefinition	56
\@Glsdescplural@: added redefinition	57
\@Glsfirst@: added redefinition	54
\@Glsfirstplural@: added redefinition	55
\@Glsname@: added redefinition	56
\@Glsplural@: added redefinition	54
\@Glsymbol@: added redefinition	57
\@Glsymbolplural@: added redefinition	58
\@Glsxtr@defaultnoglossarywarning: new	111
\@Glsxtr@field@linkdefs: new	52
\@Glsxtr@insertdots: new	154
\@print@glossary: added redefinition	108
\glsabbrvdefaultfont: renamed from \abbrvdefaultfont	159
\glsaccessdesc: new	123
\glsaccessdescplural: new	124
\glsaccessfirst: new	121
\glsaccessfirstplural: new	122
\glsaccesslong: new	126
\glsaccesslong: new	126
\glsaccessname: new	119
\glsaccessplural: new	121
\glsaccessshort: new	125
\glsaccessshort: new	125
\glsaccessshortpl: new	125

\glsaccessshortpl: new	125	\cGLSpl: new	87
\glsaccesssymbol: new	122	\cGLSpl@: new	87
\glsaccesssymbolplural: new	123	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	120	new	82
\glsentryfmt: added check for short	47	\cGLS: new	86
\glslongpltok: new	154	\cGLSformat: new	87
\glsshortpltok: new	154	\cGLSpl: new	87
\glsxtr@newabbreviation: fixed family name in \setkeys	155	\cGLSplformat: new	87
\glsxtrdiscardperiod: added check for plural	151	\GlossariesExtraWarningNoLine:	
\GLSxtrlongpl: new	169	new	14
\Glsxtrlongpl: new	168	\glsenableentrycount: new	82
\glsxtrlongpl: new	168	\glsfirstabbrvdefaultfont: new	159
\glsxtrNoGlossaryWarning: new	18	\glsfirstlongdefaultfont: new	159
\glsxtrpostlinkAddDescOnFirstUse: new	150	\Glsfmtfirst: new	278
\glsxtrpostlinkAddSymbolOnFirstUse: new	150	\glsfmtfirst: new	277
\glsxtrpostlinkendsentence: new	150	\Glsfmtfirstpl: new	278
\GLSxtrshortpl: new	167	\glsfmtfirstpl: new	278
\Glsxtrshortpl: new	166	\Glsfmtplural: new	277
\glsxtrshortpl: new	166	\glsfmtplural: new	277
short-long-desc: fixed name to use		\Glsfmtshort: changed to use	
\glslabeltok	179	\Glsxtrtitleshort	276
long-short-desc: fixed name to use		renamed from \Glsentryfmtshort	276
\glslabeltok	177	\glsfmtshort: changed to use	
0.4 (2015-12-03)		\glsxtrtitleshort	275
\glsxtr@doabbreviationsdef: added redefinition of \acronymtype	14	renamed from \Glsentryfmtshort	275
\Glsfmtshort: changed to use		\Glsfmtshortpl: changed to use	
\Glsxtrshort	276	\Glsxtrtitleshortpl	276
\glsfmtshort: changed to use		renamed from	
\glsxtrshort	275	\Glsentryfmtshortpl	276
\Glsfmtshortpl: changed to use		\glsfmtshortpl: changed to use	
\glsxtrshortpl	276	\glsxtrtitleshortpl	275
\glsfmtshortpl: changed to use		renamed from	
\glsxtrshortpl	275	\Glsentryfmtshortpl	275
\glsxtrifemptyglossary: new	21	\glsfmttext: new	276
\glsxtrnewnumber: added extra argument	135	\glsfmttext: new	276
\glsxtrnewsymbol: added extra argument	134	\glshasattribute: new	131
\MakeAcronymsAbbreviations: set the default type to \acronymtype	95	\glshascategoryattribute: new	131
\newterm: fixed name argument	134	\glsxtremsuffix: new	217
0.5 (2015-12-07)		\GlsXtrEnableEntryCounting: new	81
\cGLS: new	86	\glsxtrifcounttrigger: new	84
\cGLS@: new	87	\glsxtrscfont: new	190
		\glsxtrscsuffix: new	190
		\glsxtrsmfont: new	204
		\glsxtrsmsuffix: new	204
		short-em: new	224
		short-em-desc: new	226
		short-em-footnote: new	234
		short-em-long: new	221
		short-em-long-desc: new	222

short-em-postfootnote: new	236	\glxtrheadshortpl: now uses headuc attribute	267
short-sc-footnote: new	200	\Glsxtrheadtext: now uses headuc attribute	269
short-sc-postfootnote: new	202	\glxtrheadtext: now uses headuc attribute	268
short-sm: new	208	short-em-footnote: switch off regular attribute if set	235
short-sm-desc: new	209	short-long: switch off regular attribute if set	178
short-sm-footnote: new	214	short-long-desc: switch off regular attribute if set	179
short-sm-long: new	206	short-sc-footnote: switch off regular attribute if set	200
short-sm-long-desc: new	207	short-sm-footnote: switch off regular attribute if set	214
short-sm-postfootnote: new	215	long-short: switch off regular attribute if set	176
long-noshort-em: new	227	long-short-desc: switch off regular attribute if set	177
long-noshort-em-desc: new	231	long-short-sc-desc: switch off regular attribute if set	192
long-noshort-sm: new	210	footnote: switch off regular attribute if set	180
long-noshort-sm-desc: new	212	postfootnote: switch off regular attribute if set	182
long-short-em: new	218	0.5.2 (2015-12-08)	
long-short-em-desc: new	219	\@GLSdesc@: added accessibility support	56
long-short-sm: new	204	\@GLSdescplural@: added accessibility support	57
long-short-sm-desc: new	205	\@GLSfirst@: added accessibility support	54
0.5.1 (2015-12-02)		\@GLSfirstplural@: added accessibility support	55
\Glsaccessstext: new	120	\@GLSname@: added accessibility support	56
0.5.1 (2015-12-07)		\@GLSplural@: added accessibility support	55
\@glxtr@doaccsupp: new	17	\@GLSsymbol@: added accessibility support	57
General: removed \ifglxtruseuhead	267	\@GLSsymbolplural@: added accessibility support	58
\Glsaccessdesc: new	124	\@GLStext@: added accessibility support	53
\Glsaccessdescplural: new	124	\@GLSdesc@: added accessibility support	56
\Glsaccessfirst: new	121	\@GLSdescplural@: added accessibility support	57
\Glsaccessfirstplural: new	122	\@GLSfirst@: added accessibility support	54
\Glsaccessname: new	120	\@GLSfirstplural@: added accessibility support	55
\Glsaccessplural: new	121	\@GLSname@: added accessibility support	56
\Glsaccesssymbol: new	122	\@GLSplural@: added accessibility support	55
\Glsaccesssymbolplural: new	123	\@GLSsymbol@: added accessibility support	57
\Glsxtrheadfirst: now uses headuc attribute	270	\@GLSsymbolplural@: added accessibility support	58
\glxtrheadfirst: now uses headuc attribute	270	\@GLStext@: added accessibility support	53
\Glsxtrheadfirstplural: now uses headuc attribute	271	\@GLSdesc@: added accessibility support	56
\glxtrheadfirstplural: now uses headuc attribute	271	\@GLSdescplural@: added accessibility support	57
\Glsxtrheadplural: now uses headuc attribute	270	\@GLSfirst@: added accessibility support	54
\glxtrheadplural: now uses headuc attribute	269	\@GLSfirstplural@: added accessibility support	55
\Glsxtrheadshort: now uses headuc attribute	267		
\glxtrheadshort: now uses headuc attribute	267		
\Glsxtrheadshortpl: now uses headuc attribute	268		

<code>\@Glsname@</code> : add accessibility support ..	56	<code>\GLSaccesssymbolplural</code> : new ..	123, 128
<code>\@Glsplural@</code> : added accessibility support	54	<code>\GLSaccessstext</code> : new	120, 127
<code>\@Glsymbol@</code> : added accessibility support	57	<code>\glsentryfmt</code> : moved	
<code>\@Glsymbolplural@</code> : added accessibility support	58	<code>\glssetabbrvfmt</code> from	
<code>\@Glstext@</code> : added accessibility support	53	<code>\glsxtrabbrvfmt</code> to here	47
<code>\@glsdesc@</code> : added accessibility support	56	<code>\GlsXtrEnableInitialTagging</code> : new	147
<code>\@glsdescplural@</code> : added accessibility support	57	<code>\glsxtrfieldtitlecase</code> : new	135
<code>\@glsfirst@</code> : added accessibility support	54	<code>\GlsXtrFormatLocationList</code> : new ...	45
<code>\@glsfirstplural@</code> : added accessibility support	55	<code>\glsxtrnewabbrevpresetkeyhook</code> :	
<code>\@glsname@</code> : added accessibility support	56	new	157
<code>\@glsplural@</code> : added accessibility support	54	<code>\glsxtrtagfont</code> : new	148
<code>\@glsymbol@</code> : added accessibility support	57	<code>\KV@printgloss@nonumberlist</code> : added	47
<code>\@glsymbolplural@</code> : added accessibility support	58	<code>\mfu@checkword@do</code> : added	148
<code>\@Glstext@</code> : added accessibility support	53	<code>\setabbreviationstyle</code> : added check	
<code>\@glsxtr@activate@initialtagging</code> :		for post-definition style switch	172
new	148	0.5.3 (2015-12-09)	
<code>\@glsxtr@do@titlecaps@warn</code> : new ..	148	<code>\@glsxtr@autoindex@at</code> : new	144
<code>\@glsxtr@tag</code> : new	148	<code>\@glsxtr@autoindex@encap</code> : new ...	144
General: fixed typo in glossaries-accsupp and tidied up code to use just one		<code>\@glsxtr@autoindex@esc</code> : new	145
<code>\@ifpackageloaded</code>	119	<code>\@glsxtr@autoindex@level</code> : new ...	145
removed <code>\glsxtrabbrvfmt</code>	169	<code>\@glsxtr@autoindex@setname</code> : new ..	143
<code>\glossaryentrynumbers</code> : added	45	<code>\@glsxtr@doabbreviationsdef</code> : new ..	14
<code>\Glossentrydesc</code> : added	146	General: removed	
<code>\Glossentryname</code> : added	140	<code>\GlsXtrNoGlsWarningNoAutoMakeMain</code>	
<code>\Glossentrysymbol</code> : added	147	110
<code>\glossentrysymbol</code> : added	146	<code>\glsdescwidth</code> : added	44
<code>\GLSaccessdesc</code> : new	124, 129	<code>\glspagelistwidth</code> : added	44
<code>\GLSaccessdescplural</code> : new ...	124, 129	<code>\glsxtrdoautoindexname</code> : new	142
<code>\GLSaccessfirst</code> : new	122, 128	<code>\glsxtrpostnamehook</code> : new	141
<code>\GLSaccessfirstplural</code> : new ..	122, 128	<code>\if@glsxtr@format@override</code> : new ..	141
<code>\GLSaccesslong</code> : new	126, 130	<code>\ProvidesGlossariesExtraLang</code> : new	281
<code>\GLSaccesslongpl</code> : new	126, 130	<code>\RequireGlossariesExtraLang</code> : new	281
<code>\Glsaccesslongpl</code> : new	126	0.5.4 (2015-12-15)	
<code>\glsaccesslongpl</code> : new	126	<code>\@newglossaryentry@defunitcounters</code> :	
<code>\GLSaccessname</code> : new	120, 127	new	88
<code>\GLSaccessplural</code> : new	121, 127	<code>\@GLSxtr@p@acrlong@</code> : new	75
<code>\GLSaccessshort</code> : new	125, 129	<code>\@GLSxtr@p@acrlongpl@</code> : new	75
<code>\GLSaccessshortpl</code> : new	126, 129	<code>\@GLSxtr@p@acrshort@</code> : new	75
<code>\GLSaccesssymbol</code> : new	123, 128	<code>\@GLSxtr@p@acrshortpl@</code> : new	75
		<code>\@GLSxtr@p@long@</code> : new	74
		<code>\@GLSxtr@p@longpl@</code> : new	75
		<code>\@GLSxtr@p@plural@</code> : new	73
		<code>\@GLSxtr@p@short@</code> : new	74
		<code>\@GLSxtr@p@shortpl@</code> : new	74
		<code>\@GLSxtr@p@text@</code> : new	73
		<code>\@GlsXtrEnableOnTheFly</code> : new	41
		<code>\@Glsxtr</code> : new	41
		<code>\@Glsxtr@p@acrlong@</code> : new	75
		<code>\@Glsxtr@p@acrlongpl@</code> : new	75

\@Glsxtr@p@acrshort@: new	75	\glsenableentryunitcount: new	90
\@Glsxtr@p@acrshortpl@: new	75	\glshasattribute: added check for entry's existence	131
\@Glsxtr@p@long@: new	74	\glusifattribute: added check for entry's existence	132
\@Glsxtr@p@longpl@: new	74	\glspostlinkhook: added existence check	150
\@Glsxtr@p@plural@: new	73	\Glsxtr: new	41
\@Glsxtr@p@short@: new	73	\glxtr: new	41
\@Glsxtr@p@shortpl@: new	74	\glxtrcat: new	41
\@Glsxtr@p@text@: new	73	\glxtrdowrglossaryhook: new	69
\@Glsxtrpl: new	42	\GlsXtrEnableEntryUnitCounting: new	93
\@alt@glshyp@opt: new	70	\GlsXtrEnableOnTheFly: new	40
\@glshalt@hyp@opt: new	69	\Glsxtrpl: new	42
\@glshalt@hyp@opt@char: new	70	\glxtrpl: new	42
\@glshalt@hyp@opt@keys: new	70	\glxtrpostlocalreset: new	81
\@glshincrement@currunitcount: new	89	\glxtrpostlocalunset: new	81
\@glshlocal@increment@currunitcount: new	89	\glxtrpostreset: new	81
\@glshsetdefault@glshlink@opts: new	67	\glxtrpostunset: new	81
\@glshxtr: new	41	\glxtrprotectlinks: new	72
\@glshxtr@addunitcounter: new	88	\GlsXtrSetAltModifier: new	70
\@glshxtr@currunitcount: new	90	\GlsXtrSetDefaultGlsOpts: new	68
\@glshxtr@ifunitcounter: new	88	\glxtrstarflywarn: new	40
\@glshxtr@p@acrlong@: new	75	\GlsXtrWarning: new	42
\@glshxtr@p@acrlongpl@: new	75	\MakeAcronymsAbbreviations: now disables \setacronymstyle	95
\@glshxtr@p@acrshort@: new	75	1.0 (2016-01-24)	
\@glshxtr@p@acrshortpl@: new	75	\@glshxtr@autoindexcrossrefs: new	13
\@glshxtr@p@long@: new	74	\@glshxtr@idx@displaynumberlist: new	102
\@glshxtr@p@longpl@: new	74	\@glshxtr@idx@entrynumberlist: new	104
\@glshxtr@p@plural@: new	73	\@glshxtr@noidx@displaynumberlist: new	102
\@glshxtr@p@short@: new	73	\@glshxtr@noidx@entrynumberlist: new	103
\@glshxtr@p@shortpl@: new	74	\@glshxtr@noidx@numberlistloop: new	103
\@glshxtr@p@text@: new	73	\@glshxtr@reg@glosslist: new	96
\@glshxtr@prevunitcount: new	90	\makeglossaries: new	97
\@glshxtr@setentryunitcountunsetattr: new	93	1.01 (2016-02-02)	
\@glshxtr@unitcountlist: new	88	\glxtrdiscardperiod: added check for first use	151
\@glshxtrpl: new	42	short-desc: fixed typo in \glxtrinlinefullformat and added missing second argument	185
\@newglossaryentryposthook: added empty see value if not set and added 'see' to field key map	33	1.02 (2016-04-25)	
\@sGlsXtrEnableOnTheFly: new	40	\@glshxtr@current@style: new	43
\cGlsformat: added	87	\Glsfmtfull: new	280
\cglshformat: added	87		
\cGlsplformat: added	88		
\cglshplformat: added	87		
\glshdisablehyper: added	72		
\glshdohyperlink: added	70		
\glshdonohyperlink: added	72		

<code>\glsfmtfull</code> : new	280	<code>\@GLSdescplural@</code> : set abbreviation and regular format	57
<code>\Glsfmtfullpl</code> : new	280	<code>\@GLSfirst@</code> : set abbreviation format ..	54
<code>\glsfmtfullpl</code> : new	280	<code>\@GLSfirstplural@</code> : set abbreviation and regular format	55
<code>\Glsfmtlong</code> : new	279	<code>\@GLSname@</code> : set abbreviation and regular format	56
<code>\glsfmtlong</code> : new	278	<code>\@GLSplural@</code> : set abbreviation and regular format	55
<code>\Glsfmtlongpl</code> : new	279	<code>\@GLSsymbol@</code> : set regular format	57
<code>\glsfmtlongpl</code> : new	279	<code>\@GLSsymbolplural@</code> : set regular format	58
<code>\Glsxtrheadfull</code> : new	274	<code>\@GLStext@</code> : set abbreviation and regular format	53
<code>\glsxtrheadfull</code> : new	273	<code>\@GLSuseri@</code> : set regular format	58
<code>\Glsxtrheadfullpl</code> : new	275	<code>\@GLSuserii@</code> : set regular format	59
<code>\glsxtrheadfullpl</code> : new	274	<code>\@GLSuseriii@</code> : set regular format	59
<code>\Glsxtrheadlong</code> : new	272	<code>\@GLSuseriv@</code> : set regular format	59
<code>\glsxtrheadlong</code> : new	272	<code>\@GLSuseriv@</code> : set regular format	60
<code>\Glsxtrheadlongpl</code> : new	273	<code>\@GLSuservi@</code> : set regular format	60
<code>\glsxtrheadlongpl</code> : new	272	<code>\@Glsdesc@</code> : set abbreviation and regular format	56
<code>\Glsxtrtitlefull</code> : new	274	<code>\@Glsdescplural@</code> : set abbreviation and regular format	57
<code>\glsxtrtitlefull</code> : new	274	<code>\@Glsfirst@</code> : set abbreviation and regular format	54
<code>\Glsxtrtitlefullpl</code> : new	275	<code>\@Glsfirstplural@</code> : set abbreviation and regular format	55
<code>\glsxtrtitlefullpl</code> : new	274	<code>\@Glsname@</code> : set abbreviation and regular format	56
<code>\Glsxtrtitlelong</code> : new	273	<code>\@Glsplural@</code> : set abbreviation and regular format	54
<code>\glsxtrtitlelong</code> : new	272	<code>\@Glsymbol@</code> : set regular format	57
<code>\Glsxtrtitlelongpl</code> : new	273	<code>\@Glsymbolplural@</code> : set regular format	58
<code>\glsxtrtitlelongpl</code> : new	272	<code>\@Glstext@</code> : set abbreviation and regular format	53
<code>\ifglsxtrinsertinside</code> : new	175	<code>\@Glsuseri@</code> : set regular format	58
postfootnote: added redef of <code>\glsxtrsetupfulldefs</code>	182	<code>\@Glsuserii@</code> : set regular format	58
stylemods: new	18	<code>\@Glsuseriii@</code> : set regular format	59
1.03 (2016-04-27)		<code>\@Glsuseriv@</code> : set regular format	59
<code>\@GLSfirstplural@</code> : bug fix: misspelt cs name	55	<code>\@Glsuseriv@</code> : set regular format	60
<code>\@GLSplural@</code> : fixed bug <code>\@GLSplural@</code> should be redefined not <code>\@GLSplural</code>	55	<code>\@Glsdesc@</code> : set abbreviation and regular format	56
<code>\@Glsfirstplural@</code> : bug fix: misspelt cs name	55	<code>\@Glsdescplural@</code> : set abbreviation and regular format	57
<code>\@Glsplural@</code> : fixed bug <code>\@Glsplural@</code> should be redefined not <code>\@Glsplural</code>	54	<code>\@Glsfirst@</code> : set abbreviation and regular format	54
<code>\@glsplural@</code> : fixed bug <code>\@glsplural@</code> should be redefined not <code>\@glsplural</code>	54	<code>\@Glsfirstplural@</code> : set abbreviation and regular format	55
<code>\glsxtrtitlelongpl</code> : bug fix: changed <code>\glsxtrlong</code> to <code>\glsxtrlongpl</code> ..	272	<code>\@Glsname@</code> : set abbreviation and regular format	56
<code>\glsxtrtitleshortpl</code> : bug fix: changed <code>\glsxtrshort</code> to <code>\glsxtrshortpl</code>	267	<code>\@Glsplural@</code> : set abbreviation and regular format	54
1.04 (2015-04-30)		<code>\@Glsymbol@</code> : set regular format	57
short-em-footnote: renamed from "footnote-em"	234	<code>\@Glsymbolplural@</code> : set regular format	58
1.04 (2016-05-02)		<code>\@Glstext@</code> : set abbreviation and regular format	53
<code>\@@glsxtrpostloctag</code> : new	46	<code>\@Glsuseri@</code> : set regular format	58
<code>\@GLSdesc@</code> : set abbreviation and regular format	56	<code>\@Glsuserii@</code> : set regular format	58
		<code>\@Glsuseriii@</code> : set regular format	59
		<code>\@Glsuseriv@</code> : set regular format	59
		<code>\@Glsuseriv@</code> : set regular format	59
		<code>\@Glsuservi@</code> : set regular format	60
		<code>\@gls@preglossaryhook</code> : added check for entry's existence	149
		<code>\@Glsdesc@</code> : set abbreviation and regular format	56
		<code>\@glsdescplural@</code> : set abbreviation and regular format	57
		<code>\@glsfirst@</code> : set abbreviation and regular format	54

<code>\@glsfirstplural@</code> : set abbreviation and regular format	55	<code>\glxtruserparen</code> : new	238
<code>\@glsname@</code> : set abbreviation and regular format	56	<code>\glxtrusersuffix</code> : new	239
<code>\@glsplural@</code> : set abbreviation and regular format	54	<code>\GlsXtrWarnDeprecatedAbbrStyle</code> : new	174
<code>\@glsymbol@</code> : set regular format	57	short-em-long-em: new	223
<code>\@glsymbolplural@</code> : set regular format	58	short-em-long-em-desc: new	224
<code>\@glstext@</code> : set abbreviation and regular format	53	short-em-nolong: new	226
<code>\@glxtr@deprecated@abbrstyle</code> : new	174	short-em-nolong-desc: new	227
<code>\@glxtr@do@style</code> : new	19	short-em-postfootnote: renamed from “postfootnote-em”	236
<code>\@glxtr@doloctag</code> : new	47	short-footnote: new	181
<code>\@glxtr@idx@entrynumberlist</code> : switched from <code>\let</code> to <code>\newcommand</code>	104	short-long-user: new	245
<code>\@glxtr@pagetag</code> : new	46	short-long-user-desc: new	246
<code>\@glxtr@pagetag</code> : new	46	short-nolong: new	185
<code>\@glxtr@preloctag</code> : new	47	short-nolong-desc: new	187
<code>\@glxtr@postloctag</code> : new	47	short-postfootnote: new	183
<code>\@glxtr@preloctag</code> : new	46	short-sc-footnote: renamed from “footnote-sc”	200
<code>\glossentrydesc</code> : added glossdescfont attribute check	136	short-sc-nolong: new	195
<code>\Glossentryname</code> : added glossnamefont attribute check	140	short-sc-nolong-desc: new	197
<code>\glossentryname</code> : added glossnamefont attribute check	138	short-sc-postfootnote: renamed from “postfootnote-sc”	202
moved post name hook inside condition	140	short-sm-footnote: renamed from “footnote-sm”	214
<code>\glsabbrvemfont</code> : new	217	short-sm-nolong: new	209
<code>\glsabbrvuserfont</code> : new	238	short-sm-nolong-desc: new	210
<code>\glsfirstabbrvemfont</code> : new	217	short-sm-postfootnote: renamed from “postfootnote-sm”	215
<code>\glsfirstabbrvuserfont</code> : new	239	<code>\letabbreviationstyle</code> : new	174
<code>\glsfirstlongemfont</code> : new	217	<code>\newabbreviationstyle</code> : bug fix: corrected test for existence	173
<code>\glsfirstlonguserfont</code> : new	239	long-em-noshort-em: new	229
<code>\glsifnotregularcategory</code> : new	132	long-em-noshort-em-desc: new	232
<code>\glslongdefaultfont</code> : new	159	long-em-short-em: new	219
<code>\glslongemfont</code> : new	217	long-em-short-em-desc: new	220
<code>\glslongfont</code> : new	159	long-noshort: new	190
<code>\glslonguserfont</code> : new	239	long-noshort-desc: new	189
<code>\glxtrassignfieldfont</code> : new	53	long-noshort-em: renamed from “long-em”	227
<code>\GlsXtrEnablePreLocationTag</code> : new	45	long-noshort-em-desc: renamed from “long-desc-em”	231
<code>\glxtrfirstscfont</code> : new	190	long-noshort-sc: renamed from “long-sc”	197
<code>\glxtrfirstsmfont</code> : new	204	long-noshort-sc-desc: renamed from “long-desc-sc”	199
<code>\glxtrlongshortdescsort</code> : new	177	long-noshort-sm: renamed from “long-sm”	210
<code>\glxtrpostnamehook</code> : added category check	141	long-noshort-sm-desc: renamed from <code>\long-desc-sm</code>	212
<code>\glxtrregularfont</code> : new	48		
<code>\glxtruserfield</code> : new	238		

long-short-user: new	239	docdef option changed to choice	12
long-short-user-desc: new	244	\glstr@usesee: new	34
\renewabbreviationstyle: new	174	\glstrusesee: new	33
style: new	19	\glstruseseeformat: new	34
1.05 (2016-06-10)		\if@glstrdocdefrestricted: new	13
\eglssetwidest: new	290	1.07 (2016-08-15)	
\glsFindWidestAnyName: new	292	\@@glstrp: new	76
\glsFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new	297	nohyperfirst attribute	54
\glsFindWidestAnyNameSymbol: new	295	\@GLSfirstplural@: added check for	
\glsFindWidestAnyNameSymbolLocation:		nohyperfirst attribute	55
new	296	\@GLSxtrp: new	77
\glsFindWidestLevelTwo: new	293	\@Glsfirst@: added check for	
\glsFindWidestUsedAnyName: new	291	nohyperfirst attribute	54
\glsFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new	297	nohyperfirst attribute	55
\glsFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new	76
new	294	\@gls@preglossaryhook: added	
\glsFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts	149
new	295	\@glsfirst@: added check for	
\glsFindWidestUsedLevelTwo: new	292	nohyperfirst attribute	54
\glsFindWidestUsedTopLevelName:		\@glsfirstplural@: added check for	
new	291	nohyperfirst attribute	55
\glsfirstlongfootnotefont: new	179	\@glsxtrinmark: new	265
\glsgetwidestname: new	290	\@glsxtrnotinmark: new	265
\glsgetwidestsubname: new	291	\@glstrp: new	76
\glslongfootnotefont: new	179	\@glstrp@opt: new	75
\glstrAltTreeIndent: new	290	\glossxtrsetpopts: new	76
\glstralmtreeInit: new	290	\glsps: new	78
\glstralmtreePar: new	290	\glspt: new	78
\glstralmtreeSetHangIndent: new	298	\glstr@entry@p: new	77
\glstralmtreeSetSubHangIndent:		\glstrabbrvfootnote: new	180
new	298	\glstrchecknohyperfirst: new	53
\glstralmtreeSubSymbolDescLocation:		\glstrfieldtitlecasescs: new	136
new	290	\glstrifinmark: new	265
\glstralmtreeSymbolDescLocation:		\GLSxtrp: new	79
new	289	\Glsxtrp: new	78
\glstrComputeTreeIndent: new	298	\glstrp: new	77
\glstrComputeTreeSubIndent: new	298	\glstrsetpopts: new	76
\glstrtreetopindent: new	290	short-long-desc: added text key	179
short-em-long: fixed incorrect font used		fixed misspelling of \glsabbrvfont in	
by long form	221	plural key	179
\xglsssetwidest: new	290	long-short-desc: added missing text	
1.06 (2016-06-18)		key	177
\@glsdoifexistsorwarn: new	13	fixed misspelling of \glsabbrvfont	177
\@glstr@docdefval: new	12	footnote: changed first forms to use	
\@glstr@usesee: new	34	\glsfirstlongfootnotefont	180
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document	21	from first keys	182

switched from \glsfirstlongfont to \glsfirstlongfootnotefont ...	183	1.10 (2016-12-17)	\@GLSpl@: fixed bug caused by typo in command name	49
\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse	96	1.11 (2017-01-19)	\@glsxtr@do@redef@forglentries: new	6
1.08 (2016-12-13)			\@glsxtr@noidx@do: new	118
\@glsxtr@record: new	7		\@glsxtr@redef@forglentries: new ..	6
\@GLS@: added \@glsxtr@record	49		\@glsxtr@shortcutsval: new	16
\@GLSpl@: added \@glsxtr@record ...	49		\@glsxtr@unsrt@getgrouptitle: new	118
\@Gls@: added \@glsxtr@record	49		\@print@noidx@glossary: added redefinition	105
\@Glspl@: added \@glsxtr@record ...	49		\glsxtr@addloclistfield: added group key	11
\@gls@: added \@glsxtr@record	48		added location key	10
\@gls@@link@: added \@glsxtr@record	50		\glsxtr@fields: new	113
\@gls@field@link: added \@glsxtr@record	48		\glsxtr@linkprefix: new	114
\@gls@saveentrycounter: new	21		\glsxtr@org@newignoredglossary: new	28
\@glsdisp: added \@glsxtr@record ..	49		\glsxtr@s@newignoredglossary: new	29
\@glspl@: added \@glsxtr@record ...	49		\glsxtr@shortcutsval: new	113
\@glsxtr@dorecord: new	9		\glsxtr@texencoding: new	113
\@glsxtr@err@undefaction: new	6		\glsxtr@writefields: new	114
\@glsxtr@record: new	7		\GlsXtrLoadResources: new	113
\@glsxtr@warn@onexistsordo: new ...	6		\glsxtrresourcefile: changed extension to .glstex	112
\@glsxtr@warn@undefaction: new	6		\newignoredglossary: added starred version	28
\@print@unsrt@glossary: new	116	1.12 (2017-02-03)	\@@glsxtr@recordcounter: new	9
General: added record package option ...	11		\@gls@preglossaryhook: check for definition	149
\glsadd: added \@glsxtr@record	52		\@glsxtr@counterrecordhook: new ..	115
\glsdoifexists: now defines \glslabel	32		\@glsxtr@display@loc: new	106
\glsxtr@do@wrglossary: new	21		\@glsxtr@docounterrecord: new ...	115
\glsxtr@addloclistfield: new	10		\@glsxtr@longnewglossaryentry: new	28
\glsxtr@indexonly@saveentrycounter: new	10		\@glsxtr@noop@recordcounter: new ..	9
\glsxtr@record: new	115		\@glsxtr@op@recordcounter: new ...	10
\glsxtr@resource: new	113		\@glsxtr@provide@storagekey: new ..	22
\glsxtr@saveentrycounter: new	21		\@glsxtr@s@longnewglossaryentry: new	27
\glsxtr@setup@record: new	10		\@glsxtrentryfmt: new	23
\glsxtrassignfieldfont: added check for existence	53		\@glsxtrindexaliased: new	68
\glsxtrresourcefile: new	112		\@glsxtrsetaliasnoindex: new	68
\printunsrtglossaries: new	116		\@newglossaryentryposthook: added check for alias key	37
\printunsrtglossary: new	116		\@no@glsxtrindexaliased: new	68
1.09 (2016-12-16)			\@printunsrtglossary: new	116
\@glsxtr@gettype: new	102			
\@glsxtr@mixed@assign@sortkey: new	102			
\@printglossary: redefined to save options	101			
\glsxtr@makeglossaries: new	102			

General: added target key to printgloss family	101	\GlsXtrLoadResources: removed restriction on only one per document	113
\apptoglossarypreamble: new	27	\glxtrlocrangefmt: new	107
\csGlsXtrLetField: new	26	\glxtrpostlongdescription: new ..	28
\eGlsXtrSetField: new	26	\glxtrprovidestoragekey: new	22
\gGlsXtrSetField: new	26	\GlsXtrRecordCounter: new	115
\glsdohyperlink: added check for alias field	71	\glxtrresourcecount: new	113
\glsnoidxdisplayloc: added redefinition	106	\glxtrresourcefile: added catcode change for @	113
\glssettoctitle: added patch	29	\glxtrsetaliasnoindex: new	67
\glxtr@counterrecord: new	115	\GlsXtrSetField: new	25
\glxtr@langtag: new	113	\glxtrsetfieldifexists: new	25
\glxtr@newabbreviation: new	155	\glxtrunsrtdo: new	118
\glxtr@org@newignoredglossary: Added check for existence	28	\GlsXtrusefield: new	25
\glxtr@pluralsuffixes: new	113	\glxtrusefield: new	25
\glxtr@provideignoredglossary: new	30	short-postlong-user: new	242
\glxtr@s@newignoredglossary: Added check for existence	29	short-postlong-user-desc: new ...	244
\glxtr@s@provideignoredglossary: new	31	\longnewglossaryentry: added starred version	27
\glxtrabbrvpluralsuffix: new ...	160	long-postshort-user: new	240
\glxtralias: new	37	long-postshort-user-desc: new ...	242
\glxtrcopytoglossary: new	31	postdot: new	14
\glxtrdeffield: new	25	\pretoglossarypreamble: new	27
\glxtrdisplayendloc: new	107	\print@noop@unsrtglossaryunit: new	118
\glxtrdisplayendlochook: new ...	107	\print@op@unsrtglossaryunit: new	117
\glxtrdisplaysingleloc: new	106	\printunsrtglossary: added starred form	116
\glxtrdisplaystartloc: new	106	\printunsrtglossaryhandler: new .	117
\glxtrreffield: new	25	\printunsrtglossaryunit: new	10
\glxtrreentryfmt: new	23	\printunsrtglossaryunitsetup: new	117
\glxtrfielddolistloop: new	24	\provideignoredglossary: new	30
\glxtrfieldforlistloop: new	24	\s@glxtr@provide@storagekey: new	22
\glxtrfielddifylist: new	24	\s@printunsrtglossary: new	116
\glxtrfieldlistadd: new	24	\xGlsXtrSetField: new	26
\glxtrfieldlisteadd: new	24	1.13 (2017-02-07)	
\glxtrfieldlistgadd: new	24	\@glsdisp: removed	
\glxtrfieldlistxadd: new	24	\@glxtr@org@glsdisp	49
\glxtrfieldxifylist: new	24	\glxtrsetaliasnoindex: switched to \providecommand	67
\glxtrfmt: new	23	1.14 (2017-04-18)	
\GlsXtrFmtDefaultOptions: new	23	\@gls@link: added redefinition	51
\GlsXtrFmtField: new	23	\@gls@noidx@getgrouptitle: new ..	104
\glxtrifkeydefined: new	21	\@gls@removespaces: new	107
\glxtrindexaliased: new	68	\@glxtr@do@automake@err: new ...	115
\GlsXtrLetField: new	26	\@glxtr@org@gloautosee: new	19
\GlsXtrLetFieldToField: new	26	\@glxtr@record: added third arg	7
		\@glxtr@recordsee: new	10

General: added \glsadd option	1.16 (2017-06-15)
theHvalue	\@glo@autosee: added redefinition 20
added \glsadd option thevalue	\@gls@noidx@getgrouptitle: fixed
\glsdisablehyper: added redefinition .	bug
\glsenableentrycount: fixed	\@glsxtr@addunusedxrefs: added
assignment of \@cGls@	check for seealso field
\glsenableentryunitcount: fixed	\@glsxtr@dorecordnodefer: new
assignment of \@cGls@	\@glsxtr@noidx@do: use \csuse instead
\glsnavigation: new	of \csname
\glsxtr@org@getgrouptitle: new ..	\@print@unsrt@glossary: corrected
\glsxtr@recordsee: new	misspelt command
\glsxtr@writefields: added check for	\@printunsrt@glossary@handler:
automake	new
\glsxtrdisplayendloc: added check	General: added check for
for empty format	\@gls@setupsort@none
\glsxtrgetgrouptitle: new	\gls@checkseeallowed: added
\glsxtrnitwrgloss: new	redefinition
\glsxtrlocationhyperlink: new ...	\glsxtr@writefields: added
\glsxtrsetgrouptitle: new	\providecommand lines
\glsxtrsupphypernumber: new	\glsxtrautoindex: new
\ifglsxtrwrglossbefore: new	\glsxtrautoindexentry: new
1.15 (2017-05-10)	\glsxtrautoindexsort: new
\@glsxtr@dorecord: corrected	\glsxtrindexseealso: new
premature expansion of \@glslocref	\glsxtrseealsolabels: new
short-em-long-em: fixed spelling of	\glsxtrseelist: new
\glsabbrvfont	\glsxtruseseealso: new
short-long: fixed spelling of	\glsxtruseseealsoformat: new
\glsabbrvfont	\seealso: new
short-long-user: fixed spelling of	autoseeindex: new
\glsabbrvfont	1.17 (2017-08-09)
short-postlong-user: fixed spelling of	\@glsxtr@mark@wordseps: new
\glsabbrvfont	\@glsxtr@markwordseps: new
short-postlong-user-desc: fixed	\@glsxtr@noidx@displaynumberlist:
spelling of \glsabbrvfont	replace hard-coded ?? with
long-em-short-em: fixed spelling of	\glsxtrundeftag
\glsabbrvfont	\@glsxtr@noidx@entrynumberlist:
long-postshort-user: fixed spelling of	replace hard-coded ?? with
\glsabbrvfont	\glsxtrundeftag
long-postshort-user-desc: fixed	\@glsxtr@noidx@numberlistloop:
spelling of \glsabbrvfont	replace hard-coded ?? with
long-short: fixed spelling of	\glsxtrundeftag
\glsabbrvfont	\@glsxtrifhyphenstart: new
long-short-user: fixed spelling of	General: removed some inconsistencies
\glsabbrvfont	in the abbreviation styles
footnote: fixed spelling of	\glsabbrvhyphenfont: new
\glsabbrvfont	\glsabbrvonlyfont: new
postfootnote: fixed spelling of	\glsabbrvscfont: new
\glsabbrvfont	\glsabbrvsmfont: new

<code>\glsabbrvuserfont</code> : initialised to default font	238	<code>short-long-user-desc</code> : corrected first forms	246
<code>\glsfirstabbrvhyphenfont</code> : new	248	<code>short-nolong-desc-noreg</code> : new	187
<code>\glsfirstabbrvonlyfont</code> : new	261	<code>short-nolong-noreg</code> : new	185
<code>\glsfirstabbrvscfont</code> : new	190	<code>long-em-noshort-em-desc-noreg</code> : new	234
<code>\glsfirstabbrvsmfont</code> : new	204	<code>long-em-noshort-em-noreg</code> : new	230
<code>\glsfirstlonghyphenfont</code> : new	248	<code>long-hyphen-noshort-desc-noreg</code> : new	250
<code>\glsfirstlongonlyfont</code> : new	261	<code>long-hyphen-postshort-hyphen</code> : new	253
<code>\glslonghyphenfont</code> : new	248	<code>long-hyphen-postshort-hyphen-desc</code> : new	255
<code>\glslongonlyfont</code> : new	261	<code>long-hyphen-short-hyphen</code> : new	248
<code>\glslonguserfont</code> : initialised to default font	239	<code>long-hyphen-short-hyphen-desc</code> : new	249
<code>\glsxtr@newabbreviation</code> : added <code>\glsxtrorgshort</code> and <code>\glsxtrorglong</code>	155	<code>long-noshort-desc-noreg</code> : new	189
<code>\GlsXtrDefineAcShortcuts</code> : new	15	<code>long-noshort-noreg</code> : new	190
<code>\glsxtrgenabbrvfmt</code> : added check for <code>\ifglsxtrinsertinside</code>	169	<code>long-only-short-only</code> : new	261
<code>\glsxtrhyphensuffix</code> : new	248	<code>long-only-short-only-desc</code> : new	263
<code>\glsxtrifhyphenstart</code> : new	247	<code>long-short-user-desc</code> : corrected first forms	245
<code>\glsxtrlonghyphen</code> : new	252	1.18 (2017-08-10)	
<code>\glsxtrlonghyphennoshort</code> : new	250	<code>stylemods</code> : changed default value to "default"	18
<code>\glsxtrlonghyphenshort</code> : new	247	1.19 (2017-09-09)	
<code>\glsxtrlongshortdescname</code> : new	177	<code>\@glsxtr@defaultnumberformat</code> : new	7
<code>\glsxtronlydescname</code> : new	263	<code>\@glsxtr@dorecord</code> : Use <code>\@glsrecordlocref</code> instead of <code>\@glslocref</code>	9
<code>\glsxtronlydescsort</code> : new	263	<code>\@glsxtr@dorecordnodefer</code> : Use <code>\theglentrycounter</code> for the location rather than <code>\@glslocref</code>	9
<code>\glsxtronlysuffix</code> : new	261	<code>\@glsxtr@record@setting</code> : new	11
<code>\glsxtrparen</code> : new	158	<code>\@glsxtr@record@setting@alsoindex</code> : new	11
<code>\glsxtrposthyphenlong</code> : new	258	<code>\@glsxtrifhasfield</code> : new	25
<code>\glsxtrposthyphenshort</code> : new	253	General: added <code>\glslink</code> option <code>theHvalue</code>	51
<code>\glsxtrposthyphensubsequent</code> : new	253	added <code>\glslink</code> option <code>thevalue</code>	50
<code>\glsxtrshortdescname</code> : new	185	<code>\glsxtr@writefields</code> : removed double-quotes around <code>\jobname</code>	115
<code>\glsxtrshorthyphen</code> : new	258	<code>\glsxtrdoautoindexname</code> : changed format test	142
<code>\glsxtrshorthyphenlong</code> : new	256	<code>\glsxtrhyperlink</code> : new	71
<code>\glsxtrshortlongdescname</code> : new	179	<code>\glsxtrifhasfield</code> : new	25
<code>\glsxtrshortlongdescsort</code> : new	179	<code>\GlsXtrSetDefaultNumberFormat</code> : new	7
<code>\Glsxtrsubsequentfmt</code> : new	172	<code>\s@glsxtrifhasfield</code> : new	25
<code>\glsxtrsubsequentfmt</code> : new	171		
<code>\Glsxtrsubsequentplfmt</code> : new	172		
<code>\glsxtrsubsequentplfmt</code> : new	172		
<code>\glsxtrword</code> : new	155		
<code>\glsxtrwordsep</code> : new	154		
<code>short-hyphen-long-hyphen</code> : new	256		
<code>short-hyphen-long-hyphen-desc</code> : new	257		
<code>short-hyphen-postlong-hyphen</code> : new	258		
<code>short-hyphen-postlong-hyphen-desc</code> : new	260		

1.20 (2017-09-11)

\@glxtrhypernameprefix: new 101

\glsohypertarget: added redefinition 102

\printunstrtglossaryunitsetup:

switched from redefining

\glolinkprefix to

\@glxtrhypernameprefix 117

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	
\.	150, 289
\@	113
\@cGLS@	83, 92
\@cGLSpl@	83, 92
\@cGLs@	83, 91
\@cGLspl@	83, 92
\@cgls@	83, 91
\@cglspl@	83, 85, 91
\@do@wrglossary	8, 98
\@do@wrglossary	11, 12, 21, 52, 68
\@glo@assign@sortkey	102
\@glo@list	6
\@glo@type	116
\@gls@expand@field	22
\@glslocalreset	81
\@glslocalunset	81
\@glsreset	81
\@glsunset	81
\@glsxtr@autoindex@escspch	144–146
\@glsxtr@checkspch	143, 144, 146
\@glsxtr@disabledflycommand	43
\@glsxtr@record	12
\@glsxtr@recordcounter	11, 12, 115
\@glsxtrp	76, 77
\@glsxtrpostloctag	45
\@glsxtrpreloctag	45, 46
\@newglossaryentry@defcounters	82
\@newglossaryentry@defunitcounters	90
\@par	290
\@ACRlong	73
\@ACRlongpl	73
\@ACRshort	73
\@ACRshortpl	73
\@Acrlong	73
\@Acrlongpl	73
\@Acrshort	73
\@Acrshortpl	73
\@GLS@	72, 86, 87
\@GLSdesc@	57
\@GLSpl@	72, 86, 87
\@GLSplural@	73
\@GLSsymbol@	58
\@GLStext@	73
\@GLSxtr@full	161
\@GLSxtr@fullpl	162
\@GLSxtrp@acrlong@	73
\@GLSxtrp@acrlongpl@	73
\@GLSxtrp@acrshort@	73
\@GLSxtrp@acrshortpl@	73
\@GLSxtrp@long@	72
\@GLSxtrp@longpl@	72
\@GLSxtrp@plural@	72
\@GLSxtrp@short@	72
\@GLSxtrp@shortpl@	72
\@GLSxtrp@text@	72
\@GLSxtrlong	72, 165, 166
\@GLSxtrlongpl	72, 169
\@GLSxtrp	80
\@GLSxtrshort	72, 164
\@GLSxtrshortpl	72, 167
\@Gls@	72, 85, 86
\@Gls@acentryname	94
\@Gls@entry@field	65, 79
\@Gls@entryname	94
\@GlsXtrEnableOnTheFly	40
\@Glspl@	72, 85, 86
\@Glsplural@	73
\@Glstext@	73
\@Glsxtr	41, 43
\@Glsxtr@full	160
\@Glsxtr@fullpl	162
\@Glsxtrp@acrlong@	73
\@Glsxtrp@acrlongpl@	73

<code>\@Glsxtr@p@acrshort@</code>	73	<code>\@glo@alias</code>	36, 37
<code>\@Glsxtr@p@acrshortpl@</code>	73	<code>\@glo@assign@sortkey</code>	99
<code>\@Glsxtr@p@long@</code>	72	<code>\@glo@autosee</code>	19
<code>\@Glsxtr@p@longpl@</code>	72	<code>\@glo@autoseehook</code>	36
<code>\@Glsxtr@p@plural@</code>	72	<code>\@glo@category</code>	88
<code>\@Glsxtr@p@short@</code>	72	<code>\@glo@check@sortallowed</code>	100
<code>\@Glsxtr@p@shortpl@</code>	72	<code>\@glo@counterprefix</code>	9, 107
<code>\@Glsxtr@p@text@</code>	72	<code>\@glo@countunit</code>	88
<code>\@Glsxtrlong</code>	72, 165	<code>\@glo@default@sorttype</code>	99
<code>\@Glsxtrlongpl</code>	72, 168	<code>\@glo@desc</code>	28
<code>\@Glsxtrp</code>	79	<code>\@glo@descplural</code>	28
<code>\@Glsxtrpl</code>	42, 43	<code>\@glo@group</code>	11
<code>\@Glsxtrshort</code>	72, 163	<code>\@glo@label</code>	10, 11, 22, 33, 36–38, 64, 71, 291–298
<code>\@Glsxtrshortpl</code>	72, 167	<code>\@glo@location</code>	10, 11
<code>\@acrlong</code>	73	<code>\@glo@loclist</code>	10
<code>\@acrlongpl</code>	73	<code>\@glo@name</code>	142, 143
<code>\@acrshort</code>	73	<code>\@glo@no@assign@sortkey</code>	102
<code>\@acrshortpl</code>	73	<code>\@glo@parent</code>	292–294
<code>\@alt@gls@hyp@opt</code>	70	<code>\@glo@see</code>	33, 34, 37, 38
<code>\@auxout</code>	9, 10, 47, 84, 92, 97, 98, 108, 109, 112, 114, 115	<code>\@glo@seealso</code>	36
<code>\@bibgls@restreat</code>	113	<code>\@glo@sort</code>	143
<code>\@cGLS</code>	86	<code>\@glo@sorttype</code>	99, 105
<code>\@cGLS@</code>	83, 86, 92	<code>\@glo@text</code>	49
<code>\@cGLSpl</code>	87	<code>\@glo@thislettergrp</code>	119
<code>\@cGLSpl@</code>	83, 87, 92	<code>\@glo@thisvalue</code>	238
<code>\@cGls@</code>	83, 91	<code>\@glo@tmp</code>	22, 34, 64
<code>\@cGlspl@</code>	83, 92	<code>\@glo@type</code>	38, 94, 97, 100–102, 105, 106, 108, 109, 111, 112, 116, 117
<code>\@cgls@</code>	83, 91	<code>\@glo@types</code>	133, 134, 291–297
<code>\@cglspl@</code>	83, 91	<code>\@glossary@default@style</code>	43, 44, 100, 300
<code>\@disable@onlypremakeg</code>	97	<code>\@glossarystyle</code>	100, 101
<code>\@do@auxoutstuff</code>	108, 109	<code>\@gls@</code>	72, 85, 86
<code>\@do@gls@getcounterprefix</code>	9	<code>\@gls@@link</code>	50
<code>\@do@glssee</code>	36, 37	<code>\@gls@ReturnAfterFi</code>	107
<code>\@do@newglossaryentry</code>	94, 95, 157	<code>\@gls@actualchar</code>	143
<code>\@do@seeglossary</code>	11, 12, 19, 97	<code>\@gls@adjustmode</code>	52
<code>\@do@wrglossary</code>	51, 52	<code>\@gls@alt@hyp@opt</code>	70
<code>\@empty</code>	52, 60–64, 143, 144, 160–169	<code>\@gls@alt@hyp@opt@char</code>	70
<code>\@end@glsxtr@addunused</code>	38	<code>\@gls@alt@hyp@opt@keys</code>	70
<code>\@end@glsxtr@gettype</code>	99, 102	<code>\@gls@automake</code>	100
<code>\@end@glsxtr@usesees</code>	33, 34	<code>\@gls@between</code>	105
<code>\@end@glsxtrifhyphenstart</code>	247	<code>\@gls@checkedmkidx</code>	143, 144, 146
<code>\@endfortrue</code>	172	<code>\@gls@checkmkidxchars</code>	35, 143
<code>\@firstofone</code>	53, 136, 137, 142, 148	<code>\@gls@codepage</code>	109
<code>\@firstofthree</code>	49, 52, 60–63, 69, 70, 160, 162, 163, 165, 166, 168	<code>\@gls@counter</code>	8, 9, 51, 52, 68
<code>\@firstoftwo</code>	54, 55, 57, 58, 61–64, 66, 70, 96, 152, 153, 160, 162, 166–169, 265	<code>\@gls@currentlettergroup</code>	105, 116, 119
<code>\@for</code>	6, 18, 38, 82, 94, 97, 100, 105, 116, 135, 147	<code>\@gls@declareoption</code>	5
		<code>\@gls@default@longpl</code>	156, 157

<code>\@gls@doautomake</code>	100, 115	<code>\@gls@setsort</code>	51
<code>\@gls@doautomake@err</code>	115	<code>\@gls@setupsort@none</code>	12
<code>\@gls@encapchar</code>	144	<code>\@gls@short</code>	156, 157
<code>\@gls@entry@count</code>	83, 84	<code>\@gls@shortpl</code>	154, 156, 157
<code>\@gls@entry@field</code>	22, 25, 36, 64, 77–80, 83	<code>\@gls@sort</code>	118
<code>\@gls@entry@unitcount</code>	92	<code>\@gls@tmp</code>	105
<code>\@gls@field@font</code>	53–60	<code>\@gls@tmpb</code>	146
<code>\@gls@field@link</code>	53–60, 65, 66	<code>\@gls@type</code>	97, 98, 100, 172, 291–297
<code>\@gls@getcounterprefix</code>	9	<code>\@gls@write@entrycounts</code>	83
<code>\@gls@getgrouptitle</code>	104, 105, 116	<code>\@gls@write@entryunitcounts</code>	92
<code>\@gls@grptitle</code>	105	<code>\@gls@write@entryunitcounts@do</code>	93
<code>\@gls@hyp@opt</code>	65, 66, 70, 86, 87, 160–169	<code>\@gls@xref</code>	10, 35
<code>\@gls@hyp@opt@cs</code>	69, 70	<code>\@glsabbrv@current@abbreviation</code>	155, 169
<code>\@gls@increment@currcount</code>	83	<code>\@glsacronymlists</code>	94
<code>\@gls@increment@currunitcount</code>	91	<code>\@glsdoifexistsorwarn</code>	13, 138–141
<code>\@gls@keymap</code>	10, 11, 22, 33, 36, 64, 114	<code>\@glsentry</code>	84, 92, 93
<code>\@gls@label</code>	8, 9, 69, 97, 98, 115, 172	<code>\@glslink</code>	51, 71, 72
<code>\@gls@levelchar</code>	144	<code>\@glsnextpages</code>	101
<code>\@gls@link</code>	23, 48–50, 60–64, 160–169	<code>\@glsnonextpages</code>	101
<code>\@gls@link@checkfirsthyper</code>	49, 96	<code>\@glsnumberformat</code>	7, 9, 51, 52, 68, 141, 142
<code>\@gls@link@label</code>	51	<code>\@glsorder</code>	97
<code>\@gls@link@nocheckfirsthyper</code>	48, 60–64, 160–169	<code>\@glspl@</code>	72, 85, 86
<code>\@gls@link@opts</code>	51	<code>\@glsplural@</code>	73
<code>\@gls@list</code>	105	<code>\@glspunc@token</code>	152
<code>\@gls@local@increment@currcount</code>	83	<code>\@glsrecordlocref</code>	9
<code>\@gls@local@increment@currunitcount</code>	91	<code>\@glsshowtarget</code>	71
<code>\@gls@location</code>	118, 119	<code>\@glsstyle@almtree</code>	289
<code>\@gls@loclist</code>	102, 103, 118, 119	<code>\@glsstyle@inline</code>	289
<code>\@gls@long</code>	156	<code>\@glsstyle@listdotted</code>	284
<code>\@gls@longpl</code>	154, 156, 157	<code>\@glstarget</code>	72, 101
<code>\@gls@nohyperlist</code>	29–31	<code>\@glstext@</code>	73
<code>\@gls@noidx@do</code>	106	<code>\@glswidestname</code>	290, 291, 298
<code>\@gls@noidx@getgrouptitle</code>	116	<code>\@glsxtr</code>	41, 43
<code>\@gls@noidx@nosanitizesort</code>	99	<code>\@glsxtr@do@wrglossary</code>	98
<code>\@gls@noidx@sanitizesort</code>	99	<code>\@glsxtr@abbreviationsdef</code>	14, 20
<code>\@gls@noidx@loclist@finalsep</code>	102, 103	<code>\@glsxtr@activate@initialtagging</code>	147, 149
<code>\@gls@noidx@loclist@prev</code>	102, 103	<code>\@glsxtr@addunitcounter</code>	88
<code>\@gls@noidx@loclist@sep</code>	102	<code>\@glsxtr@addunused</code>	38
<code>\@gls@noref@warn</code>	98, 106	<code>\@glsxtr@addunusedxrefs</code>	38
<code>\@gls@org@glsnoidx@displayloc</code>	103	<code>\@glsxtr@attrval</code>	136–142
<code>\@gls@org@glsseeformat</code>	103	<code>\@glsxtr@autoindex@at</code>	143, 144
<code>\@gls@preglossaryhook</code>	101, 147	<code>\@glsxtr@autoindex@doextra@esc</code>	143
<code>\@gls@prevlevel</code>	299	<code>\@glsxtr@autoindex@encap</code>	142, 144
<code>\@gls@quotechar</code>	143	<code>\@glsxtr@autoindex@esc</code>	143, 145, 146
<code>\@gls@reference</code>	39, 97, 98	<code>\@glsxtr@autoindex@escat</code>	143, 144
<code>\@gls@saveentrycounter</code>	11, 12, 21, 51, 52	<code>\@glsxtr@autoindex@escencap</code>	144, 145
<code>\@gls@see@noindex</code>	20, 113	<code>\@glsxtr@autoindex@esclevel</code>	144, 145
<code>\@gls@setdefault@glslink@opts</code>	51, 68	<code>\@glsxtr@autoindex@escquote</code>	143, 145

<code>\@glsxtr@autoindex@level</code>	144, 145	<code>\@glsxtr@format@overridefalse</code>	141
<code>\@glsxtr@autoindex@setname</code>	142	<code>\@glsxtr@format@overridetrue</code>	142
<code>\@glsxtr@autoindexcrossrefs</code>	12, 13, 33, 36	<code>\@glsxtr@foundinlist</code>	152
<code>\@glsxtr@autoseeindexfalse</code>	12	<code>\@glsxtr@full</code>	160
<code>\@glsxtr@autoseeindextrue</code>	13	<code>\@glsxtr@fullpl</code>	161
<code>\@glsxtr@cat</code>	82, 94, 147	<code>\@glsxtr@gettype</code>	99
<code>\@glsxtr@counterrecordhook</code>	9, 10	<code>\@glsxtr@glossdescfont</code>	136, 137
<code>\@glsxtr@csname</code>	89, 91	<code>\@glsxtr@glossnamefont</code>	138–141
<code>\@glsxtr@current@style</code>	43, 300	<code>\@glsxtr@gobbleto@endescspch</code>	146
<code>\@glsxtr@currentunitcount</code>	89, 91	<code>\@glsxtr@idx@displaynumberlist</code>	98
<code>\@glsxtr@currunitcount</code>	90, 92	<code>\@glsxtr@idx@entrynumberlist</code>	99
<code>\@glsxtr@declareoption</code>	5, 14, 18	<code>\@glsxtr@ifcsstart</code>	40
<code>\@glsxtr@defaultnoglossarywarning</code>	18	<code>\@glsxtr@ifpunctoken</code>	152
<code>\@glsxtr@defaultnumberformat</code>	7, 51, 52, 68, 141, 142	<code>\@glsxtr@ifunitcounter</code>	88
<code>\@glsxtr@deprecated@abbrstyle</code>	199, 200, 202, 204, 212, 214, 215, 217, 229, 232, 236, 238	<code>\@glsxtr@insert@dots</code>	154
<code>\@glsxtr@disabledflycommand</code>	43	<code>\@glsxtr@insert@dots@next</code>	154
<code>\@glsxtr@display@loc</code>	106	<code>\@glsxtr@insertdots</code>	156
<code>\@glsxtr@do@@wrindex</code>	69	<code>\@glsxtr@label</code>	38, 135
<code>\@glsxtr@do@@glsdisablehyperinlist</code>	67	<code>\@glsxtr@loadstyles</code>	283
<code>\@glsxtr@do@@redef@for@gl@sentries</code>	7	<code>\@glsxtr@longnewglossaryentry</code>	27
<code>\@glsxtr@do@@style</code>	19, 282	<code>\@glsxtr@mark@wordseps</code>	155
<code>\@glsxtr@do@@titlecaps@warn</code>	136–139, 148	<code>\@glsxtr@mark@wordseps@next</code>	155
<code>\@glsxtr@do@abbreviationsdef</code>	14	<code>\@glsxtr@mark@wordseps</code>	156, 157
<code>\@glsxtr@do@accsupp</code>	18, 19	<code>\@glsxtr@mixed@assign@sortkey</code>	99
<code>\@glsxtr@docdefval</code>	12, 13, 39	<code>\@glsxtr@noidx@displaynumberlist</code>	98
<code>\@glsxtr@docounterrecord</code>	10	<code>\@glsxtr@noidx@do</code>	118
<code>\@glsxtr@doglossary</code>	116, 117	<code>\@glsxtr@noidx@entrynumberlist</code>	99
<code>\@glsxtr@doloctag</code>	46	<code>\@glsxtr@noidx@numberlistloop</code>	99
<code>\@glsxtr@dorecord</code>	8	<code>\@glsxtr@noop@recordcounter</code>	9, 11
<code>\@glsxtr@dorecordnodefer</code>	8	<code>\@glsxtr@notfoundinlist</code>	152
<code>\@glsxtr@dostylewarn</code>	172	<code>\@glsxtr@op@recordcounter</code>	12
<code>\@glsxtr@enabletagging</code>	147	<code>\@glsxtr@optlist</code>	42
<code>\@glsxtr@end@</code>	40	<code>\@glsxtr@org@GLS@</code>	49
<code>\@glsxtr@endescspch</code>	143–146	<code>\@glsxtr@org@GLSpl@</code>	49
<code>\@glsxtr@entrycount@org@localreset</code>	83	<code>\@glsxtr@org@Gls@</code>	49
<code>\@glsxtr@entrycount@org@localunset</code>	83	<code>\@glsxtr@org@Glspl@</code>	49
<code>\@glsxtr@entrycount@org@reset</code>	83	<code>\@glsxtr@org@Glsxtr@titlefirst</code>	265, 266
<code>\@glsxtr@entrycount@org@unset</code>	83	<code>\@glsxtr@org@Glsxtr@titlefirstplural</code>	265, 266
<code>\@glsxtr@entryunitcount@org@localreset</code>	91	<code>\@glsxtr@org@Glsxtr@titlefull</code>	265, 266
<code>\@glsxtr@entryunitcount@org@localunset</code>	91	<code>\@glsxtr@org@Glsxtr@titlefullpl</code>	265, 266
<code>\@glsxtr@entryunitcount@org@reset</code>	91	<code>\@glsxtr@org@Glsxtr@titlelong</code>	265, 266
<code>\@glsxtr@entryunitcount@org@unset</code>	91	<code>\@glsxtr@org@Glsxtr@titlelongpl</code>	265, 266
<code>\@glsxtr@err@undefaction</code>	7, 11	<code>\@glsxtr@org@Glsxtr@titleplural</code>	265, 266
<code>\@glsxtr@field@linkdefs</code>	48	<code>\@glsxtr@org@Glsxtr@titleshort</code>	265, 266
		<code>\@glsxtr@org@Glsxtr@titleshortpl</code>	265, 266
		<code>\@glsxtr@org@Glsxtr@titletext</code>	265, 266
		<code>\@glsxtr@org@MakeUppercase</code>	265, 266
		<code>\@glsxtr@org@checkfirsthyper</code>	66, 96

<code>\@glsxtr@org@delimN</code>	46	<code>\@glsxtr@recordsee</code>	12
<code>\@glsxtr@org@delimR</code>	46	<code>\@glsxtr@redef@for@gl@sentries</code>	7, 20
<code>\@glsxtr@org@doseeglossary</code>	11, 12, 98	<code>\@glsxtr@redefstyles</code>	18, 19, 282
<code>\@glsxtr@org@gloautosee</code>	20	<code>\@glsxtr@reg@glosslist</code>	97–100, 102
<code>\@glsxtr@org@gls@</code>	48	<code>\@glsxtr@s@longnewglossaryentry</code>	27
<code>\@glsxtr@org@glsdohypertarget</code>	102	<code>\@glsxtr@savepreloctag</code>	45, 47
<code>\@glsxtr@org@glsignore</code>	46	<code>\@glsxtr@setentrycountunsetattr</code>	82
<code>\@glsxtr@org@glspl@</code>	49	<code>\@glsxtr@setentryunitcountunsetattr</code>	93
<code>\@glsxtr@org@glsxtrtitlefirst</code>	265, 266	<code>\@glsxtr@setupshortcuts</code>	17, 20
<code>\@glsxtr@org@glsxtrtitlefirstplural</code>	265, 266	<code>\@glsxtr@shortcutsval</code>	17, 114
<code>\@glsxtr@org@glsxtrtitlefull</code>	265, 266	<code>\@glsxtr@swaptwo</code>	153
<code>\@glsxtr@org@glsxtrtitlefullpl</code>	265, 266	<code>\@glsxtr@tag</code>	147
<code>\@glsxtr@org@glsxtrtitlelong</code>	265, 266	<code>\@glsxtr@taggingcs</code>	147
<code>\@glsxtr@org@glsxtrtitlelongpl</code>	265, 266	<code>\@glsxtr@theHvalue</code>	7, 8, 51, 52
<code>\@glsxtr@org@glsxtrtitleplural</code>	265, 266	<code>\@glsxtr@thevalue</code>	7, 8, 50–52
<code>\@glsxtr@org@glsxtrtitleshort</code>	265, 266	<code>\@glsxtr@thisloctag</code>	46, 47
<code>\@glsxtr@org@glsxtrtitleshortpl</code>	265, 266	<code>\@glsxtr@titlelabel</code>	104, 105, 118
<code>\@glsxtr@org@glsxtrtitletext</code>	265, 266	<code>\@glsxtr@tmp</code>	18, 107
<code>\@glsxtr@org@makeglossaries</code>	97	<code>\@glsxtr@type</code>	135
<code>\@glsxtr@org@markboth</code>	264, 265	<code>\@glsxtr@unitcountlist</code>	88
<code>\@glsxtr@org@markright</code>	264, 265	<code>\@glsxtr@unsrt@getgrouptitle</code>	116
<code>\@glsxtr@org@newacronymstyle</code>	95, 96	<code>\@glsxtr@usesee</code>	34
<code>\@glsxtr@org@postdescription</code>	149	<code>\@glsxtr@warn@onexistsordo</code>	7, 12
<code>\@glsxtr@org@see@noindex</code>	113	<code>\@glsxtr@warn@undefaction</code>	7, 12
<code>\@glsxtr@org@setacronymstyle</code>	95, 96	<code>\@glsxtr@docdeffalse</code>	39
<code>\@glsxtr@org@theHvalue</code>	7, 8	<code>\@glsxtr@entryfmt</code>	23
<code>\@glsxtr@org@prefix</code>	9	<code>\@glsxtr@hypernameprefix</code>	101, 102, 117
<code>\@glsxtr@org@printglossary</code>	43, 101	<code>\@glsxtr@trifhasfield</code>	25
<code>\@glsxtr@org@warndep</code>	153	<code>\@glsxtr@trifhyphenstart</code>	247
<code>\@glsxtr@p@acrlong@</code>	73	<code>\@glsxtr@indexaliased</code>	68
<code>\@glsxtr@p@acrlongpl@</code>	73	<code>\@glsxtr@indexcrossrefsfalse</code>	13
<code>\@glsxtr@p@acrshort@</code>	73	<code>\@glsxtr@indexcrossrefstrue</code>	13
<code>\@glsxtr@p@acrshortpl@</code>	73	<code>\@glsxtr@inmark</code>	264
<code>\@glsxtr@p@long@</code>	72	<code>\@glsxtr@long</code>	72, 164, 165
<code>\@glsxtr@p@longpl@</code>	72	<code>\@glsxtr@longpl</code>	72, 168
<code>\@glsxtr@p@plural@</code>	72	<code>\@glsxtr@notinmark</code>	264
<code>\@glsxtr@p@short@</code>	72	<code>\@glsxtr@p</code>	78
<code>\@glsxtr@p@shortpl@</code>	72	<code>\@glsxtr@p@opt</code>	76
<code>\@glsxtr@p@text@</code>	72	<code>\@glsxtr@pl</code>	42, 43
<code>\@glsxtr@pagetag</code>	45, 46	<code>\@glsxtr@postloctag</code>	45, 47
<code>\@glsxtr@pagetag</code>	45, 46	<code>\@glsxtr@preloctag</code>	45, 47
<code>\@glsxtr@prevunitcount</code>	90	<code>\@glsxtr@setaliasnoindex</code>	67, 68
<code>\@glsxtr@printglossopts</code>	43, 99, 101	<code>\@glsxtr@short</code>	72, 163
<code>\@glsxtr@provide@addstoragekey</code>	22	<code>\@glsxtr@shortpl</code>	72, 166
<code>\@glsxtr@provide@storagekey</code>	22	<code>\@glsxtr@undeftag</code>	6, 21
<code>\@glsxtr@record</code>	11, 12, 48–50, 52	<code>\@gobble</code>	7, 11, 12, 14, 53, 117, 154
<code>\@glsxtr@record@setting</code>	8, 11	<code>\@gobbletwo</code>	153
<code>\@glsxtr@record@setting@alsoindex</code>	8	<code>\@ifnextchar</code>	69, 70

<code>\@ifpackageloaded</code>	
... 5, 14, 114, 119, 136, 137, 140, 142, 281	
<code>\@ifstar</code> ...	22, 25, 27, 28, 30, 40, 69, 116, 147
<code>\@ifundefined</code>	281
<code>\@ignored@glossaries</code>	29–31
<code>\@input</code>	113
<code>\@input@</code>	108
<code>\@istfilename</code>	97
<code>\@makeglossary</code>	97
<code>\@mfu@domakefirststuc</code>	148
<code>\@mfu@nocaplist</code>	148
<code>\@ne</code>	84, 93
<code>\@newglossaryentry@defcounters</code> ..	82, 90
<code>\@newglossaryentryposthook</code>	
.....	10, 11, 22, 36, 64
<code>\@newglossaryentryprehook</code>	
.....	10, 11, 22, 27, 28, 36, 64
<code>\@nil</code>	107, 118
<code>\@nnil</code>	143, 144, 146, 152–155
<code>\@no@glxstrindexaliased</code>	68
<code>\@no@makeglossaries</code>	112
<code>\@nopostdesc</code>	101
<code>\@onelevel@sanitize</code>	10, 35, 42, 104, 105, 118
<code>\@onlypreamble</code>	
...	43, 46, 92, 113, 115, 142, 144, 145, 147
<code>\@org@glossaryentrynumbers</code>	100, 101
<code>\@org@newglossaryentryprehook</code> ...	27, 28
<code>\@print@unsrt@glossary</code>	116
<code>\@printgloss@setsort</code>	99, 100
<code>\@printglossary</code>	42, 116
<code>\@printunsrt@glossary@handler</code>	117
<code>\@printunsrtglossary</code>	116
<code>\@sGlsXtrEnableOnTheFly</code>	40
<code>\@secondofthree</code>	
.....	53–55, 61–65, 161, 162, 164, 165, 167, 168
<code>\@secondoftwo</code>	49,
.....	52, 56–64, 66, 72, 96, 101, 153, 160,
.....	161, 163–169, 182, 203, 216, 237, 265, 266
<code>\@sglsxtr@provide@storagekey</code>	22
<code>\@thirdofthree</code>	53–
.....	56, 61–64, 66, 161, 162, 164, 166, 167, 169
<code>\@thirdoftwo</code>	56–60
<code>\@warn@nomakeglossaries</code>	109
<code>\@xdy@main@language</code>	108
<code>\@xdycrossrefhook</code>	35
<code>\@xdy@language</code>	108
<code>\@xdy@locationclassorder</code>	35
<code>\@</code>	107
<code>_</code>	110, 111
A	
<code>\AB</code>	15
<code>\Ab</code>	15
<code>\ab</code>	15
abbreviation styles:	
long-hyphen-postshort-hyphen	252, 253, 255
long-hyphen-short-hyphen	249, 253
long-postshort-user	242
long-short-user	240
short	185
short-hyphen-long-hyphen	257, 258
short-hyphen-postlong-hyphen ...	258, 260
short-long-user	242
short-nolong	185
short-nolong-desc	187
short-postlong-user	244
<code>\abbreviationsname</code>	14
<code>\abbrvpluralsuffix</code>	
.....	114, 156, 176, 178, 180,
.....	182, 184, 186, 188, 191, 193, 194, 196,
.....	197, 199, 201, 203, 205, 206, 208, 209,
.....	211, 212, 214, 216, 218, 220, 221, 223,
.....	225–227, 229, 231, 233, 235, 237, 239,
.....	241, 243, 246, 249, 251, 254, 257, 259, 262
<code>\ABP</code>	15
<code>\Abp</code>	15
<code>\abp</code>	15
<code>\AC</code>	16
<code>\Ac</code>	15
<code>\ac</code>	15
<code>\ACF</code>	16
<code>\Acf</code>	16
<code>\acf</code>	15
<code>\ACFP</code>	16
<code>\Acfp</code>	16
<code>\acfp</code>	15
<code>\ACL</code>	16
<code>\Acl</code>	15
<code>\acl</code>	15
<code>\ACLP</code>	16
<code>\Aclp</code>	16
<code>\aclp</code>	15
<code>\ACP</code>	16
<code>\Acp</code>	15
<code>\acp</code>	15
<code>\ACRfullfmt</code>	95

<code>\Acryfallmt</code>	95		
<code>\acrryfallmt</code>	95		
<code>\ACRfullplfmt</code>	95		
<code>\Acrfullplfmt</code>	95		
<code>\acrfullplfmt</code>	95		
<code>\acronymentry</code>	94		
<code>\acronymfont</code>	60–64, 75, 95, 96		
<code>\acronymname</code>	14		
<code>\acronymsort</code>	94		
<code>\acronymtype</code>	14, 94, 95		
<code>\acrpluralsuffix</code>	94, 114		
<code>\ACS</code>	16		
<code>\Acs</code>	15		
<code>\acs</code>	15		
<code>\ACSP</code>	16		
<code>\Acsp</code>	15		
<code>\acsp</code>	15		
<code>\actualchar</code>	145		
<code>\addtolength</code>	299		
<code>\advance</code>	84, 93, 113		
<code>\AF</code>	15		
<code>\Af</code>	15		
<code>\af</code>	15		
<code>\AFP</code>	15		
<code>\Afp</code>	15		
<code>\afp</code>	15		
<code>\AL</code>	15		
<code>\Al</code>	15		
<code>\al</code>	15		
<code>\ALP</code>	15		
<code>\Alp</code>	15		
<code>\alp</code>	15		
<code>\AnyTrackedLanguages</code>	281		
<code>\appto</code>	10, 11, 19, 22, 33, 35–37, 64, 69, 82, 90, 117, 142, 152, 154, 155		
<code>\AS</code>	15		
<code>\As</code>	15		
<code>\as</code>	15		
<code>\ASP</code>	15		
<code>\Asp</code>	15		
<code>\asp</code>	15		
<code>\AtBeginDocument</code>	21, 44, 45, 114		
<code>\AtEndDocument</code>	38, 83, 92, 108, 109		
B			
babel package	142, 144, 152		
<code>\begin</code>	106, 110, 111, 116, 284–288		
<code>\begingroup</code>	7, 68, 116		
<code>\bgroup</code>	27, 28, 100		
C			
<code>\catcode</code>	113		
category attributes:			
apospplural	156		
discardperiod	151		
entrycount	81, 82, 84, 93		
firsttuc	140		
glossdesc	136		
glossdescfont	136		
glossname	137		
glossnamefont	138, 140		
headuc	267		
indexname	143		
indexonlyfirst	68		
insertdots	156		
markshortwords	156		
markwords	156, 157, 247, 248, 256		
nohyper	67		
nohyperfirst	53–55		
noshortplural	156		
regular	47, 87, 175– 180, 182, 184–190, 192–196, 198, 200, 202, 206, 207, 209, 210, 212–214, 216, 219–225, 227, 228, 230–232, 234–236, 239, 245, 247, 249, 250, 252, 257, 262, 263		
<code>\cGLS</code>	15, 16, 82, 93		
<code>\cGls</code>	15, 82, 93		
<code>\cgls</code>	15, 82, 93		
<code>\cGLSformat</code>	86		
<code>\cGlsformat</code>	85		
<code>\cglsformat</code>	85, 87		
<code>\cGLSpl</code>	15, 16, 82, 93		
<code>\cGlspl</code>	15, 82, 93		
<code>\cglspl</code>	15, 82, 93		
<code>\cGLSplformat</code>	86		
<code>\cGlsplformat</code>	85		
<code>\cglsplformat</code>	85, 87		
<code>\char</code>	104		
<code>\columnwidth</code>	44, 45		
<code>\count@</code>	84, 92, 93		
<code>\csappto</code>	27		
<code>\csdef</code>	22, 25, 27, 64–66, 83, 88, 89, 91, 130, 173, 174, 182, 202, 216, 236, 240, 242, 244, 254, 255, 259, 261, 284		
<code>\cseappto</code>	31		
<code>\csedef</code>	25, 89		
<code>\csgdef</code>	26, 29–31, 39, 46, 83, 89, 91, 92		
<code>\cslet</code>	26, 28, 101		
<code>\csletcs</code>	26, 174		

`\csname` 6, 8, 22, 30, 35, 43, 47, 49, 51,
 52, 60–66, 68, 76, 89, 97, 98, 105, 108,
 111, 112, 117, 136, 153, 160–169, 175, 298
`\cspreto` 27
`\csuse` 23, 30, 37, 46, 65, 66,
 77–79, 88–92, 100, 104–106, 115, 117–
 119, 130, 141, 149, 150, 173, 174, 291–294
`\csxdef` 33, 36, 89, 92, 105
`\currentglossary` 101
`\CurrentOption` 19, 283
`\CurrentTrackedLanguageTag` 114
`\CurrentTrackedTag` 281
`\CustomAbbreviationFields` 157,
 175, 177, 179, 180, 182, 183, 185, 187,
 189, 191–194, 196, 197, 200, 202, 204–
 210, 214, 215, 218–224, 226, 227, 229,
 232, 234, 236, 239, 240, 242, 244–246,
 248–250, 252, 253, 255–257, 259–261, 263

D

`\DeclareAcronymList` 94
`\DeclareOption` 5, 283
`\DeclareOptionX` 5, 19
`\def` 7–12, 21, 28, 30, 34–36, 38, 40–43, 45,
 47–64, 70–75, 85–87, 94, 98–102, 104–
 108, 116, 118, 143–148, 152–156, 160–
 169, 172, 247, 248, 250, 253, 256, 258, 299
`\defglstentryfmt` 29–31
`\define@boolkey` 13, 67
`\define@choicekey` 7, 11, 13, 16, 18, 50, 101
`\define@key` 10,
 11, 18, 19, 22, 36, 50–52, 64, 101, 153, 154
`\DefineAcronymSynonyms` 17
`\delimN` 46
`\delimR` 46
`\detokenize` 40
`\dimen@` 96, 291–298
`\dimen@i` 292–294
`\dimen@ii` 292–294
`\dimexpr` 44, 45, 290
`\disable@keys` 14, 21, 39
`\do` 6, 18, 38, 82, 94, 97, 100, 105, 117, 135, 147
`\do@gls@link@checkfirsthyper`
 23, 48–51, 60–64, 160–169
`\do@gls@disablehyperinlist` 51, 67
`doc` package 145
`\dolistcsloop` 24
`\DTLifinlist` 97–99, 102
`\DTLifint` 104

E

`\eappto` 10, 18, 29–31, 117, 142, 283
`\edef` 6, 8, 9, 29–32, 34, 36, 37, 51, 52, 66, 68,
 71, 88, 89, 91, 97, 98, 102, 104, 106–109,
 113, 136–141, 143, 144, 146, 153, 292–294
`\eglssetwidest` 291–298
`\egroup` 28, 101
`\else` 8–10, 13, 14, 16, 18, 19, 26,
 39, 40, 45, 47, 50–52, 68, 69, 84, 96, 99–
 101, 104, 106, 107, 110, 112, 113, 142–
 144, 146, 152, 154, 155, 157, 163–169,
 171, 172, 176, 178, 181, 183–189, 191–
 193, 195–205, 207–220, 222–238, 240,
 241, 243, 244, 246–248, 250, 253, 255,
 256, 258, 260, 262, 263, 285–289, 299, 300
`\emph` 217
`\empty` 106, 107
`\encapchar` 146
`\end` 106, 110, 111, 117, 284–288
`\end@glsxtr@display@loc` 106
`\endcsname` 6, 8, 22, 30, 35, 43, 47, 49, 51, 52,
 60–66, 68, 76, 89, 97, 98, 105, 108, 109,
 111, 112, 117, 136, 153, 160–169, 175, 298
`\endgroup` 8, 68, 116
 entry categories:
 abbreviation 169
 general 130, 132
 index 134
`\epreto` 143
`\equal` 111, 112
`etoolbox` package 5
`\expandafter` 19, 22,
 23, 33, 34, 38, 40–42, 65, 66, 70, 76, 87,
 88, 97–99, 102, 105, 107, 116–119, 136,
 139, 140, 142, 145, 146, 152, 156, 157, 247
`\expandonce` 94, 143, 144, 177, 250

F

`\fi` 7–10,
 12–14, 16–18, 20, 26, 33, 36, 38–40, 44,
 45, 47, 50–52, 68, 69, 84, 92, 93, 96, 99–
 101, 104, 106, 107, 109–113, 115, 142–
 144, 146, 153–155, 157, 163–169, 171,
 172, 176, 178, 180, 181, 183–189, 191–
 193, 195–205, 207–220, 222–238, 240,
 241, 243, 244, 246–248, 250, 253, 255,
 256, 258, 260, 262, 263, 285–289, 291–300
 first use 301
 flag 301
 text 301

<code>\firstacronymfont</code>	95, 96	<code>\glossarysection</code>	105, 106, 111, 116, 117
fontspec package	114	<code>\glossarytitle</code>	30, 100, 101, 105, 106, 111, 116
<code>\footnote</code>	180	<code>\glossarytoctitle</code>	30, 100, 105, 106, 111, 116
<code>\forall glossaries</code>	38, 116, 133–135, 291–297	<code>\glossentry</code>	101, 119, 284–288, 299
<code>\forall glsentries</code>	84, 93	<code>\glossentrydesc</code>	284–290
<code>\ForEachTrackedDialect</code>	281	<code>\glossentryname</code>	284–288, 299
<code>\forall glsentries</code>	6, 38, 133–135, 291–297	<code>\glossentrysymbol</code>	285–290
<code>\forall listcsloop</code>	24, 93, 106	<code>\glossxtrsetpopts</code>	149
<code>\forall listloop</code>	102, 103, 148	<code>\GLS</code>	82, 93
<code>\futurelet</code>	152	<code>\Gls</code>	41, 82, 93
G			
<code>\gdef</code>	46, 144, 145	<code>\gls</code>	26, 41, 43, 82, 93, 98, 110
<code>\Genacrfullformat</code>	95	<code>\gls@assign@desc</code>	28
<code>\genacrfullformat</code>	95	<code>\gls@assign@field</code>	10, 11, 22, 64
<code>\GenericAcronymFields</code>	95	<code>\gls@checkseeallowed</code>	40, 97
<code>\Genplacrfullformat</code>	95	<code>\gls@codepage</code>	109
<code>\genplacrfullformat</code>	95	<code>\gls@defdocnewglossaryentry</code>	82, 90
<code>\glo@grabfirst</code>	118	<code>\gls@defglossaryentry</code>	28, 41, 42
<code>\glo@name</code>	138–140	<code>\gls@dotoctitle</code>	100, 101
<code>\gloaliaslabel</code>	71	<code>\gls@glossary</code>	35
<code>\global</code>	9, 28, 101, 118, 119	<code>\gls@level</code>	118
<code>\glo linkprefix</code>	51, 71, 114	<code>\gls@noidxglossary</code>	98
glossaries package	12, 19, 20, 33, 35, 37, 100, 283	<code>\gls@org@glossaryentryfield</code>	101
glossaries-accsupp package	18, 19, 119	<code>\gls@org@glossarysubentryfield</code>	101
glossaries-extra package	2	<code>\gls@save@numberlist</code>	45, 47
glossaries-extra-stylemods package	18, 149, 282	<code>\gls@set@xr@key</code>	35, 36
glossaries.sty package	28	<code>\gls@tmplen</code>	291–297, 299
<code>\GlossariesExtraWarning</code>	6, 14, 27, 40, 42, 96, 98, 107, 110, 113, 116, 136–141, 147, 148, 174	<code>\gls@type</code>	98
<code>\GlossariesExtraWarningNoLine</code>	14, 84, 93	<code>\glsabbrvdefaultfont</code>	159, 176, 178, 180, 182, 184, 186, 188, 238, 248, 251, 261
<code>\GlossariesWarning</code>	46, 98, 100, 103, 104, 172	<code>\glsabbrvemfont</code>	217–229, 231, 233–237
<code>\GlossariesWarningNoLine</code>	98, 109	<code>\glsabbrvfont</code>	73, 74, 95, 159, 163, 164, 166, 167, 171, 172, 175–180, 182, 184–186, 188, 189, 191, 193, 194, 196, 197, 199, 201, 203, 205, 206, 208, 209, 211, 212, 214, 216, 218, 220, 222, 223, 225, 226, 228, 229, 231, 233, 235, 237, 239, 241, 243, 245–247, 249, 251–254, 257, 259, 262
glossary styles:		<code>\glsabbrvhyphenfont</code>	248, 249, 253–257, 259, 260
<code>alttree</code>	289, 290, 298	<code>\glsabbrvonlyfont</code>	261–263
<code>inline</code>	289	<code>\glsabbrvscfont</code>	190–194, 196, 197, 199–203
<code>listdotted</code>	284	<code>\glsabbrvsmfont</code>	204–212, 214–216
<code>listdottedstyle</code>	284	<code>\glsabbrvuserfont</code>	239–246
<code>sublistdotted</code>	284	<code>\GLSaccessdesc</code>	56
glossary-long package	285	<code>\Glsaccessdesc</code>	56, 136, 146
glossary-longbooktabs package	285	<code>\glsaccessdesc</code>	56, 137, 150
glossary-mcols package	283	<code>\GLSaccessdescplural</code>	57
glossary-tree package	283	<code>\Glsaccessdescplural</code>	57
<code>\glossaryentrynumbers</code>	47, 100, 101, 118, 119	<code>\glsaccessdescplural</code>	57
<code>\glossaryheader</code>	106, 116, 284–288, 299	<code>\glsaccessdescplural</code>	57
<code>\glossaryname</code>	100		
<code>\glossarypostamble</code>	106, 117		
<code>\glossarypreamble</code>	105, 116		

<code>\GLSaccessfirst</code>	54	222, 224–228, 230, 232–237, 240, 241,
<code>\Glsaccessfirst</code>	54	243, 246, 249, 251, 254, 257, 259, 260, 263
<code>\glsaccessfirst</code>	54	<code>\GLSaccesssymbol</code>
<code>\GLSaccessfirstplural</code>	56	<code>\Glsaccesssymbol</code>
<code>\Glsaccessfirstplural</code>	55	<code>\glsaccesssymbol</code>
<code>\glsaccessfirstplural</code>	55	<code>\GLSaccesssymbolplural</code>
<code>\Glsaccesslong</code>	63, 158, 165, 176,	<code>\Glsaccesssymbolplural</code>
	184, 188, 191, 195, 198, 199, 205, 208,	<code>\glsaccesssymbolplural</code>
	211, 213, 218, 220, 225, 228–233, 240,	<code>\GLSaccessstext</code>
	241, 249, 251, 252, 254, 255, 257, 262, 263	<code>\Glsaccessstext</code>
<code>\glsaccesslong</code>	62, 63, 158, 165,	<code>\glsaccessstext</code>
	166, 176, 178, 181, 183, 184, 186, 188,	<code>\glsacrshortcutstrue</code>
	189, 191, 193, 195–203, 205, 207–215,	<code>\glsacspacemax</code>
	217, 218, 220, 222–238, 240, 241, 243,	<code>\glsadd</code>
	244, 246, 249, 251, 252, 254, 255, 257, 262	<code>\glsadd options</code>
<code>\Glsaccesslongpl</code>	64, 158, 168, 176,	<code>theHvalue</code>
	184, 188, 192, 195, 198–200, 205, 208,	<code>thevalue</code>
	211, 213, 218, 220, 225, 228, 230–234,	<code>\glsaddstoragekey</code>
	240, 241, 249, 251, 252, 255, 257, 262, 263	<code>\glsbackslash</code>
<code>\glsaccesslongpl</code>		<code>\glscapscase</code>
	63, 64, 158, 168, 169, 176, 178,	<code>\glscategory</code>
	181, 183, 184, 186, 188, 189, 191, 193, 47, 53, 66, 73, 74, 131–133, 136–
	195–205, 207, 208, 210–215, 217, 218,	141, 146, 147, 149, 150, 160–164, 166, 167
	220, 222, 224–236, 238, 240, 241, 243,	<code>\glscategorylabel</code>
	244, 246, 249, 251, 252, 254, 255, 257, 262 66, 153, 155–157, 182, 202,
<code>\GLSaccessname</code>	56	216, 236, 240, 242, 244, 254, 255, 259, 261
<code>\Glsaccessname</code>	56	<code>\glsfclosebrace</code>
<code>\glsaccessname</code>	56	<code>\glscurrententrylabel</code>
<code>\GLSaccessplural</code>	55 45–47, 101, 108, 116, 117, 148, 149
<code>\Glsaccessplural</code>	55	<code>\glscurrentfieldvalue</code>
<code>\glsaccessplural</code>	54	<code>\glscustomtext</code> ..
<code>\GLSaccessshort</code>	61,	<code>\glsdefaulttype</code> 6, 14, 27, 99, 100, 110, 116, 117
	164, 172, 178, 181, 183, 186, 193, 196,	<code>\glsdescriptionaccessdisplay</code> 123, 124, 136
	201–203, 207, 210, 215–217, 222, 224,	<code>\glsdescriptionpluralaccessdisplay</code> 124
	226, 235–238, 243, 244, 246, 254, 259, 260	<code>\glsdescwidth</code>
<code>\glsaccessshort</code>	60, 61,	<code>\glsdetoklabel</code>
	158, 163, 164, 171, 176, 178, 180, 181,	8, 24–26, 28, 32–
	183–188, 191, 193, 195–201, 203, 205,	34, 38–40, 51, 52, 68, 71, 83, 88–93, 97,
	207–211, 213–218, 220, 222, 223, 225–	101–103, 118, 119, 136, 138–140, 292–294
	228, 230, 232, 233, 235–237, 240, 241,	<code>\glsdisplaynumberlist</code>
	243, 246, 249, 251, 254, 257, 259, 260, 262	<code>\glsdohyperlink</code>
<code>\Glsaccessshortpl</code>	62, 167,	<code>\glsdohypertarget</code>
	172, 178, 181, 183, 186, 193, 196, 201–	<code>\glsdoifexists</code>
	204, 207, 210, 215–217, 222, 224, 227,	13, 25, 31,
	235–238, 243, 244, 246, 254, 259, 260, 263	33, 34, 48, 49, 52, 60–64, 103, 104, 160–169
<code>\glsaccessshortpl</code>	61, 62, 158,	<code>\glsdoifexistsordo</code>
	166, 167, 172, 176, 178, 181, 183–188,	<code>\glsdoifexistsorwarn</code> 13, 136, 137, 146, 147
	191–193, 195–201, 203, 205, 207–220,	<code>\glsdoifnoexists</code>
		<code>\glsdonohyperlink</code>
		<code>\glsdosanitizesort</code>
		<code>\glsenableentrycount</code>

<code>\glsenableentryunitcount</code>	83, 93	<code>\glsentryuseriv</code>	59
<code>\glsentrycounter</code>	107	<code>\Glsentryuserv</code>	59
<code>\glsentrycurrcount</code>	83, 84, 90	<code>\glsentryuserv</code>	60
<code>\Glsentrydesc</code>	124, 129, 137	<code>\Glsentryuservi</code>	60
<code>\glsentrydesc</code>	124, 128, 129, 137	<code>\glsentryuservi</code>	60
<code>\Glsentrydescplural</code>	124, 129	<code>\glsfieldfetch</code>	71
<code>\glsentrydescplural</code>	124, 125, 129	<code>\glsfieldxdef</code>	135
<code>\Glsentryfirst</code>	87, 121, 128	<code>\glsfindwidesttoplevelname</code>	291
<code>\glsentryfirst</code> 87, 121, 122, 127, 128, 277, 278		<code>\GLSfirst</code>	270
<code>\Glsentryfirstplural</code>	88, 122, 128	<code>\Glsfirst</code>	271
<code>\glsentryfirstplural</code>	87, 122, 128, 278	<code>\glsfirst</code>	270
<code>\glsentryfmt</code>	29–31	<code>\glsfirstabbrvdefaultfont</code>	
<code>\Glsentryfull</code>	95	159, 176, 178, 180, 182, 184, 186, 188, 251	
<code>\glsentryfull</code>	95	<code>\glsfirstabbrvemfont</code>	218–238
<code>\Glsentryfullpl</code>	95	<code>\glsfirstabbrvfont</code>	95,
<code>\glsentryfullpl</code>	95	158, 176–188, 191, 193, 194, 196, 197,	
<code>\glsentryitem</code>	284–288, 299	199, 201, 203, 205, 206, 208, 209, 211,	
<code>\Glsentrylong</code>	74, 75, 87, 126, 129	212, 214, 216, 218, 220, 222, 223, 225,	
<code>\glsentrylong</code>	74, 75, 87, 126, 129,	226, 228, 229, 231, 233, 235, 237, 239,	
130, 182, 202, 216, 237, 243, 244, 258, 279		241, 243, 246, 249, 251, 254, 257, 259, 262	
<code>\Glsentrylongpl</code>	74, 75, 88, 126, 130	<code>\glsfirstabbrvhyphenfont</code>	
<code>\glsentrylongpl</code> 74, 75, 87, 126, 127, 130, 279		248, 249, 253, 254, 256–260
<code>\Glsentryname</code>	120, 127, 138–141	<code>\glsfirstabbrvonlyfont</code>	262, 263
<code>\glsentryname</code> ...	119, 120, 127, 143, 291–298	<code>\glsfirstabbrvscfont</code>	191–204
<code>\glsentrynumberlist</code>	99, 104, 296–298	<code>\glsfirstabbrvsmfont</code>	204–217
<code>\Glsentryplural</code>	121, 127	<code>\glsfirstabbrvuserfont</code>	239–246
<code>\glsentryplural</code>	121, 127, 277	<code>\glsfirstaccessdisplay</code>	121, 122
<code>\glsentryprevcount</code>	83, 84, 90	<code>\glsfirstlongdefaultfont</code>	
<code>\glsentryprevmaxcount</code>	91	176, 178, 184, 186, 188, 191–200, 204–	
<code>\glsentryprevtotalcount</code>	90	213, 218, 219, 221, 222, 225–229, 231, 232	
<code>\Glsentryshort</code>	73, 75, 125, 129	<code>\glsfirstlongemfont</code>	
<code>\glsentryshort</code>	73–	219–221, 223, 224, 229, 230, 233, 234
75, 96, 125, 129, 240, 242, 253, 275, 276		<code>\glsfirstlongfont</code> ...	158, 175–180, 182,
<code>\Glsentryshortpl</code>	74, 75, 125, 129	184, 186–189, 191, 193, 194, 196, 197,	
<code>\glsentryshortpl</code>		199, 201, 203, 205, 206, 208, 209, 211,	
.....	74, 75, 125, 126, 129, 275, 276	212, 214, 216, 218, 220, 222, 223, 225,	
<code>\Glsentrysymbol</code>	123, 128	226, 228, 229, 231, 233, 235, 237, 239,	
<code>\glsentrysymbol</code>	122, 123, 128, 295, 296	241, 243, 246, 249, 251, 254, 257, 259, 262	
<code>\Glsentrysymbolplural</code>	123, 128	<code>\glsfirstlongfootnotefont</code>	
<code>\glsentrysymbolplural</code>	123, 128	180–183, 200–204, 214–217, 235–238
<code>\Glsentrytext</code>	120, 127	<code>\glsfirstlonghyphenfont</code>	248–259
<code>\glsentrytext</code>	71, 120, 127, 276, 277	<code>\glsfirstlongonlyfont</code>	262, 263
<code>\Glsentryuseri</code>	58	<code>\glsfirstlonguserfont</code>	239–247
<code>\glsentryuseri</code>	58	<code>\GLSfirstplural</code>	271
<code>\Glsentryuserii</code>	59	<code>\Glsfirstplural</code>	271
<code>\glsentryuserii</code>	59	<code>\glsfirstplural</code>	271
<code>\Glsentryuseriii</code>	59	<code>\glsfirstpluralaccessdisplay</code>	122
<code>\glsentryuseriii</code>	59	<code>\glsforeachincategory</code>	172
<code>\Glsentryuseriv</code>	59	<code>\glsgenentryfmt</code>	47, 48

<code>\glsgetattribute</code>	70, 71, 84, 88–90, 108, 136–142	<code>thevalue</code>	51
<code>\glsgetcategoryattribute</code>	131	<code>wrgloss</code>	50
<code>\glsgetwidestname</code>	290	<code>\glslinkcheckfirsthyperhook</code>	67
<code>\glsgroupheading</code>	119, 284–288, 299	<code>\glslinkpostsetkeys</code>	51
<code>\glsgroupskip</code>	119, 285–289, 300	<code>\glslinkvar</code>	69, 70
<code>\glsattribute</code>	70, 84, 89, 91, 93, 108, 136–142, 176–180, 182, 185, 187, 189–194, 201, 202, 204, 206, 207, 214, 216, 218–224, 231, 234, 235, 237, 239, 240, 242–247, 249, 250, 252, 254, 256–259, 261–263	<code>\glslistdottedwidth</code>	284
<code>\glsascategoryattribute</code>	131	<code>\glslocalunset</code>	50
<code>\gls hyperlink</code>	71	<code>\glslongaccessdisplay</code>	126
<code>\gls hypernavsep</code>	105	<code>\glslongdefaultfont</code>	159, 176, 178, 179, 184, 186, 188, 191, 193, 194, 196–199, 205, 206, 208, 209, 211–213, 218, 222, 225–228, 231, 239, 248, 261
<code>\gls hypernumber</code>	108, 142	<code>\glslongemfont</code>	217, 220, 223, 229, 230, 232, 233
<code>\glsifattribute</code>	50, 53, 67, 69, 77, 134, 136–139, 148, 151, 267–275	<code>\glslongfont</code>	74, 75, 159, 165, 166, 168, 169, 176–180, 182, 184, 186–189, 191, 193, 194, 196, 197, 199, 201, 203, 205, 206, 208, 209, 211, 212, 214, 216, 218, 220, 222, 223, 225, 226, 228, 229, 231, 233, 235, 237, 239, 241, 243, 246, 249, 251, 254, 257, 259, 262, 263
<code>\glsifcategory</code>	133	<code>\glslongfootnotefont</code>	179, 180, 182, 201, 203, 214, 216, 235, 237
<code>\glsifcategoryattribute</code>	66, 132, 156, 157	<code>\glslonghyphenfont</code>	248–254, 256, 257, 259
<code>\glsifnotregular</code>	53	<code>\glslongonlyfont</code>	261, 262
<code>\glsifnotregularcategory</code>	133	<code>\glslongpltok</code>	157, 176–180, 187, 189, 191–194, 197, 200, 204–207, 210, 211, 214, 218, 219, 221–224, 227, 229, 233, 235, 239, 240, 242, 244, 245, 247–250, 252, 253, 255–257, 262, 263
<code>\glsifplural</code>	49, 52, 54, 55, 57, 58, 60–64, 151, 160–170	<code>\glslongpluralaccessdisplay</code>	126
<code>\glsifregular</code>	47, 53, 87, 88	<code>\glslongtok</code>	94, 155–157, 175–180, 182, 184, 186, 187, 189, 191, 192, 194, 196, 197, 200, 202, 204–211, 214, 215, 218–227, 229, 232, 233, 235, 236, 239, 240, 242, 244–246, 248–250, 252, 253, 255–257, 259, 262, 263
<code>\glsifregularcategory</code>	133	<code>\glslonguserfont</code>	239–246
<code>\glsifusetranslator</code>	30	<code>\glsnameaccessdisplay</code> ..	119, 120, 138–140
<code>\glsignore</code>	46	<code>\glsnamefont</code>	138–141
<code>\glsinlinedescformat</code>	289	<code>\glsnavhyperlink</code>	105
<code>\glsinlinesubdescformat</code>	289	<code>\glsnextpages</code>	101
<code>\glsinsert</code>	49, 52, 60–64, 160–171, 247, 254, 255, 259, 261	<code>\glsnoidxdisplayloc</code>	103
<code>\glskeylisttok</code>	94, 95, 155, 157	<code>\glsnoidxdisplaylocclsthandler</code>	102
<code>\glslabel</code>	32, 47, 48, 50, 51, 66–68, 70, 71, 96, 150, 151, 169–171, 182, 202, 216, 236, 237, 240, 242–244, 254, 255, 259, 261	<code>\glsnoidxloclist</code>	103, 118, 119
<code>\glslabeltok</code>	94, 155, 157, 175–180, 182, 184–187, 189–194, 196, 197, 200–202, 204–209, 211, 214, 216, 218–227, 229, 231, 233–235, 237, 239, 240, 242–250, 252, 254, 256–259, 261–263	<code>\glsnoidxnumberlistloophandler</code>	103
<code>\glsletentryfield</code>	143	<code>\glsnonnextpages</code>	101
<code>\glslink</code>	95	<code>\glsnonnumberlistfalse</code>	45
<code>\glslink options</code>		<code>\glsnonnumberlisttrue</code>	45
<code>counter</code>	8	<code>\glsnumberlistloop</code>	99
<code>format</code>	141		
<code>hyper</code>	264		
<code>noindex</code>	7, 67, 264		
<code>theHvalue</code>	51		

<code>\glsnumlistlastsep</code>	102	<code>\Glstext</code>	269
<code>\glsnumlistsep</code>	102	<code>\glstext</code>	269
<code>\glsopenbrace</code>	35, 111, 112	<code>\glstextaccessdisplay</code>	120
<code>\glsorder</code>	97	<code>\glstextformat</code>	50, 51
<code>\glspagelistwidth</code>	284, 286–288	<code>\glstextup</code>	190
<code>\glspar</code>	117	<code>\glstreeindent</code>	298, 299
<code>\GLSpl</code>	82, 93	<code>\glstreenamebox</code>	299
<code>\Glspl</code>	42, 82, 93	<code>\glstreenamefmt</code>	290–299
<code>\glspl</code>	42, 82, 93	<code>\GlsrLetField</code>	26
<code>\GLSplural</code>	269, 270	<code>\glstype</code>	49, 51, 60–64, 160–169
<code>\Glsplural</code>	270	<code>\glsunset</code>	38, 50, 85, 86
<code>\glsplural</code>	269	<code>\glswrite</code>	35, 97
<code>\glspluralaccessdisplay</code>	121	<code>\glswriteentry</code>	8
<code>\glspluralsuffix</code>	114, 156, 160	<code>\Glsxtr</code>	43
<code>\glspostdescription</code>	149, 284–290	<code>\glsxtr</code>	43
<code>\glspostinline</code>	289	<code>\glsxtr@do@wrglossary</code>	8, 11, 12
<code>\glspostlinkhook</code> ..	48, 50, 60–64, 76, 160–169	<code>\glsxtr@addloclistfield</code>	12
<code>\glsprestandardsort</code>	99	<code>\glsxtr@addunused</code>	38
<code>\glsresetentrylist</code>	106, 116	<code>\glsxtr@applyabbrfmt</code>	169
<code>\glssee</code>	35, 37	<code>\glsxtr@applyabbrvstyle</code>	153, 155, 173
<code>\glsseeformat</code>	34, 98, 103	<code>\glsxtr@counterrecord</code>	115
<code>\glsseelist</code>	34	<code>\glsxtr@doption</code>	5, 14, 19, 20
<code>\glssetabbrvfmt</code>	47, 53, 73, 74, 136–141, 146, 147, 160–164, 166, 167	<code>\glsxtr@fields</code>	114
<code>\glssetattribute</code> ...	176–180, 182, 184– 187, 189–194, 196, 197, 200–202, 205– 209, 211, 214, 216, 218–221, 223–227, 229, 231, 233–235, 237, 239, 240, 242– 247, 249, 250, 252, 254, 256–259, 261–263	<code>\glsxtr@headentry@p</code>	77, 78
<code>\glssetcategoryattribute</code>	82, 94, 96, 131, 132, 134, 135, 147	<code>\glsxtr@ifnextpunc</code>	152
<code>\glssetnoexpandfield</code>	10, 11	<code>\glsxtr@ifpunctoken</code>	152
<code>\glssettoctitle</code>	100	<code>\glsxtr@indexonly@saveentrycounter</code>	11, 12, 21
<code>\glsshortaccessdisplay</code>	125	<code>\glsxtr@keylist</code>	41, 42
<code>\glsshortpltok</code> 157, 176–180, 182, 184, 186, 191–194, 196, 200, 202, 204–209, 214, 215, 218–226, 235, 236, 239, 240, 242, 244–249, 253, 255–257, 259, 260, 262, 263		<code>\glsxtr@langtag</code>	114
<code>\glsshortpluralaccessdisplay</code> ..	125, 126	<code>\glsxtr@linkprefix</code>	114
<code>\glsshorttok</code>	94, 155–157, 175–180, 182, 184–186, 189, 191, 192, 194, 196, 197, 200, 202, 204–210, 214, 215, 218, 219, 221–227, 229, 234–236, 239, 240, 242, 244–249, 252, 253, 255–257, 259–263	<code>\glsxtr@makeglossaries</code>	97
<code>\glssubentryitem</code>	284–289, 299	<code>\glsxtr@newabbreviation</code>	95, 155
<code>\glsymbolaccessdisplay</code>	122, 123	<code>\glsxtr@next</code>	152, 153
<code>\glsymbolpluralaccessdisplay</code>	123	<code>\glsxtr@org@getgrouptitle</code>	105
<code>\glstarget</code>	284–289, 299	<code>\glsxtr@org@newignoredglossary</code>	28
<code>\GLStext</code>	268, 269	<code>\glsxtr@org@makenoidxglossaries</code>	39
		<code>\glsxtr@pluralsuffixes</code>	114
		<code>\glsxtr@provideignoredglossary</code>	30
		<code>\glsxtr@punclist</code>	152
		<code>\glsxtr@record</code>	9
		<code>\glsxtr@recordsee</code>	10
		<code>\glsxtr@resource</code>	112, 114
		<code>\glsxtr@s@newignoredglossary</code>	28
		<code>\glsxtr@s@provideignoredglossary</code> ..	30
		<code>\glsxtr@saveentrycounter</code>	8, 10, 68
		<code>\glsxtr@setup@record</code>	11, 12, 20, 21
		<code>\glsxtr@shortcutsval</code>	114
		<code>\glsxtr@texencoding</code>	114

<code>\glxtr@usesee</code>	33	<code>\glxtrfieldtitlecasecs</code>	136
<code>\glxtr@warnonexistsordo</code> .	7, 11, 12, 32, 33	<code>\glxtrfieldxifinlist</code>	117
<code>\glxtr@writefields</code>	112	<code>\glxtrfirstscfont</code>	190
<code>\glxtrabbrvfootnote</code>		<code>\glxtrfirstsmfont</code>	204
....	180–182, 200–202, 214–216, 235, 236	<code>\GlsXtrFmtDefaultOptions</code>	23
<code>\glxtrabbrvpluralsuffix</code>	114,	<code>\GlsXtrFmtField</code>	23
	160, 176, 178, 180, 182, 184, 186, 188,	<code>\GlsXtrFormatLocationList</code> .	45, 47, 296–298
	190, 204, 217, 239, 248, 251, 254, 259, 261	<code>\GLSxtrfull</code>	15, 16, 273, 274
<code>\glxtrabbrvtype</code>	14, 157	<code>\Glsxtrfull</code>	15, 16, 274
<code>\glxtraddallcrossrefs</code>	38	<code>\glxtrfull</code>	15, 273, 274
<code>\glxtralias</code>	68	<code>\Glsxtrfullformat</code>	
<code>\glxtrAltTreeIndent</code>	290	158, 171, 173, 174, 176, 178, 181,
<code>\glxtralttreeInit</code>	298		183, 185, 186, 189, 191, 193, 195, 197,
<code>\glxtrAltTreePar</code>	289		198, 200, 201, 203, 205, 207, 209, 210,
<code>\glxtrAltTreeSetHangIndent</code> ...	290, 299		212–214, 216, 218, 220, 222, 224, 226,
<code>\glxtrAltTreeSetSubHangIndent</code> ...	299		227, 229, 230, 232, 234, 235, 237, 240,
<code>\glxtralttreeSubSymbolDescLocation</code> .	299		241, 243, 246, 249, 252, 254, 257, 260, 262
<code>\glxtralttreeSymbolDescLocation</code> ..		<code>\glxtrfullformat</code>	
....	290, 299	158, 171, 173, 174, 176, 178, 180,
<code>\glxtrassignfieldfont</code>	53–60		182, 184, 186, 188, 191, 193, 195, 196,
<code>\glxtrautoindex</code>	142		198, 200, 201, 203, 205, 207, 209, 210,
<code>\glxtrautoindexassignsort</code>	143		212–214, 216, 218, 220, 222, 223, 225,
<code>\glxtrautoindexentry</code>	143		227, 228, 230, 232, 234, 235, 237, 240,
<code>\glxtrcat</code>	41, 42		241, 243, 246, 249, 252, 254, 257, 260, 262
<code>\glxtrchecknohyperfirst</code>	54–56	<code>\GLSxtrfullpl</code>	15, 16, 274, 275
<code>\glxtrComputeTreeIndent</code>	299	<code>\Glsxtrfullpl</code>	15, 16, 275
<code>\glxtrComputeTreeSubIndent</code>	299	<code>\glxtrfullpl</code>	15, 274
<code>\Glsxtrdefaultsubsequentfmt</code> ...	172, 173	<code>\Glsxtrfullplformat</code>	
<code>\glxtrdefaultsubsequentfmt</code> ...	171, 173	.	159, 171, 173, 174, 176, 178, 181, 183,
<code>\Glsxtrdefaultsubsequentplfmt</code> .	172, 173		185, 187, 189, 192, 193, 195, 197, 198,
<code>\glxtrdefaultsubsequentplfmt</code> .	172, 173		200, 201, 203, 205, 207, 209, 210, 212,
<code>\GlsXtrDefineAbbreviationShortcuts</code> .	17		213, 215, 216, 218, 220, 222, 224, 226,
<code>\GlsXtrDefineAcShortcuts</code>	17		227, 229, 230, 232, 234, 235, 237, 240,
<code>\GlsXtrDefineOtherShortcuts</code>	17		241, 243, 246, 249, 252, 255, 257, 260, 262
<code>\glxtrdiscardperiod</code>	150	<code>\glxtrfullplformat</code>	
<code>\glxtrdisplayendloc</code>	106	171, 173, 174, 176, 178, 181,
<code>\glxtrdisplayendlochook</code>	107		183, 185, 186, 189, 191, 193, 195, 196,
<code>\glxtrdisplaysingleloc</code>	106, 107		198, 200, 201, 203, 205, 207, 209, 210,
<code>\glxtrdisplaystartloc</code>	106		212–214, 216, 218, 220, 222, 223, 225,
<code>\glxtrdoautoindexname</code>	69, 141		227, 229, 230, 232, 234, 235, 237, 240,
<code>\glxtrdopostpunc</code>	182, 202, 216, 236		241, 243, 246, 249, 252, 254, 257, 260, 262
<code>\glxtrdowrglossaryhook</code>	69	<code>\glxtrfullsep</code>	158, 175–179,
<code>\glxxtremsuffix</code>	218, 220,		181, 183, 184, 186, 188, 191–196, 198–
	221, 223, 225–227, 229, 231, 233, 235, 237		213, 215, 217–228, 230, 232–234, 236–
<code>\GlsXtrEnableEntryCounting</code>	93		238, 248, 249, 251, 253, 256–258, 262, 263
<code>\GlsXtrEnableEntryUnitCounting</code>	82	<code>\glxtrgenabbrvfmt</code>	48
<code>\GlsXtrEnableOnTheFly</code>	40, 43	<code>\glxtrgetgrouptitle</code>	105
<code>\glxtrfieldlistgadd</code>	115	<code>\Glsxtrheadfirst</code>	266
<code>\glxtrfieldtitlecase</code>	136–139	<code>\glxtrheadfirst</code>	266

<code>\Glsxtrheadfirstplural</code>	266	215, 217, 225, 227, 228, 230, 232, 234,
<code>\glsxtrheadfirstplural</code>	266	236, 238, 241, 244, 251, 255, 260, 263, 281
<code>\Glsxtrheadfull</code>	266	<code>\glsxtrinlinfullplformat</code>
<code>\glsxtrheadfull</code>	266 159, 162, 163, 173,
<code>\Glsxtrheadfullpl</code>	266	174, 181, 183, 184, 186, 188, 195, 196,
<code>\glsxtrheadfullpl</code>	266	198, 199, 201, 203, 208, 210, 211, 213,
<code>\Glsxtrheadlong</code>	266	215, 217, 225, 226, 228, 230, 232, 233,
<code>\glsxtrheadlong</code>	266	236, 237, 241, 243, 251, 255, 260, 262, 280
<code>\Glsxtrheadlongpl</code>	266	<code>\glsxtrininsertinsidefalse</code>
<code>\glsxtrheadlongpl</code>	266 175
<code>\Glsxtrheadplural</code>	266	<code>\glsxtrlocationhyperlink</code>
<code>\glsxtrheadplural</code>	266 107
<code>\Glsxtrheadshort</code>	266	<code>\glsxtrlocrangefmt</code>
<code>\glsxtrheadshort</code>	266 106, 107
<code>\Glsxtrheadshortpl</code>	266	<code>\GLSxtrlong</code>
<code>\glsxtrheadshortpl</code>	266 15, 16, 272
<code>\Glsxtrheadtext</code>	266	<code>\Glsxtrlong</code>
<code>\glsxtrheadtext</code>	266 15, 273
<code>\glsxtrhyperlink</code>	71, 107	<code>\glsxtrlong</code>
<code>\glsxtrhyphensuffix</code>	249, 257 15, 272
<code>\glsxtrifcounttrigger</code>	85, 86	<code>\glsxtrlonghyphen</code>
<code>\glsxtrifemptyglossary</code>	106, 111, 116 254, 255
<code>\glsxtrifhyphenstart</code>	248, 250, 253, 256, 258	<code>\glsxtrlonghyphennoshort</code>
<code>\glsxtrifindexing</code>	69 251, 252
<code>\glsxtrifinmark</code>	77–80, 265, 266	<code>\glsxtrlonghyphenshort</code>
<code>\glsxtrifnextpunc</code>	152, 153 249
<code>\glsxtrifperiod</code>	151	<code>\GLSxtrlongpl</code>
<code>\glsxtrifwasfirstuse</code> 52, 54, 55, 60–64, 66, 96, 150, 15, 16, 272, 273
..... 151, 160, 163–169, 182, 202, 203, 216, 236, 237, 240, 242, 244, 254, 255, 259, 261	<code>\Glsxtrlongpl</code>
<code>\glsxtrindexaliased</code>	68 15, 16, 273
<code>\glsxtrindexseealso</code>	36, 37	<code>\glsxtrlongpl</code>
<code>\glsxtrinitwrgloss</code>	51 15, 272
<code>\glsxtrinitwrglossbeforefalse</code>	50	<code>\glsxtrlongshortdescname</code>
<code>\glsxtrinitwrglossbeforetrue</code>	50 177, 192, 205, 219, 220, 245, 249, 255
<code>\Glsxtrinlinfullformat</code>	159, 161, 173,	<code>\glsxtrlongshortdescsort</code>
..... 174, 181, 183, 184, 186, 188, 195, 196, 177, 192, 205, 219, 220, 245, 249, 255	<code>\glsxtrmarkhook</code>
..... 198, 199, 202, 203, 208, 210, 211, 213, 264	<code>\glsxtrnewabbrevpresetkeyhook</code>
..... 215, 217, 225, 226, 228, 230, 232, 233, 157 16
..... 236, 238, 241, 244, 251, 255, 260, 263, 280	<code>\glsxtrnewnumber</code> 16
<code>\glsxtrinlinfullformat</code>	159–161, 173,	<code>\glsxtrnewsymbol</code>
..... 174, 181, 183, 184, 186, 188, 195, 196, 18, 108	<code>\GlsXtrNoGlsWarningAutoMake</code>
..... 198, 199, 201, 203, 208, 209, 211, 213, 112	<code>\GlsXtrNoGlsWarningBuildInfo</code>
..... 215, 217, 225, 226, 228, 230, 231, 233, 112	<code>\GlsXtrNoGlsWarningCheckFile</code>
..... 236, 237, 241, 243, 251, 255, 260, 262, 280 111, 112	<code>\GlsXtrNoGlsWarningEmptyMain</code> ..
<code>\Glsxtrinlinfullplformat</code>	159, 162, 173,	<code>\GlsXtrNoGlsWarningEmptyNotMain</code> ...
..... 174, 181, 183, 184, 186, 188, 195, 196, 111	<code>\GlsXtrNoGlsWarningEmptyStart</code>
..... 198, 199, 201, 203, 208, 209, 211, 213, 111	<code>\GlsXtrNoGlsWarningHead</code>
..... 215, 217, 225, 226, 228, 230, 231, 233, 112	<code>\GlsXtrNoGlsWarningMisMatch</code>
..... 236, 237, 241, 243, 251, 255, 260, 262, 280 112	<code>\GlsXtrNoGlsWarningNoOut</code>
<code>\Glsxtrinlinfullplformat</code>	159, 162, 173,	<code>\GlsXtrNoGlsWarningTail</code>
..... 174, 181, 183, 184, 186, 188, 195, 196, 112	<code>\glsxtronlydescname</code>
..... 198, 200, 202, 203, 208, 210, 211, 213, 263	<code>\glsxtronlydescsort</code>
..... 198, 200, 202, 203, 208, 210, 211, 213, 263	<code>\glsxtronlysuffix</code>
..... 198, 200, 202, 203, 208, 210, 211, 213, 262	<code>\glsxtrorg@ifKV@glslink@hyper</code>
..... 198, 200, 202, 203, 208, 210, 211, 213, 48	<code>\glsxtrorglong</code>
..... 198, 200, 202, 203, 208, 210, 211, 213, 155, 177, 250	<code>\glsxtrorgshort</code>
..... 198, 200, 202, 203, 208, 210, 211, 213, 155, 177	<code>\GLSxtrp</code>
..... 198, 200, 202, 203, 208, 210, 211, 213, 77	<code>\Glsxtrp</code>
..... 198, 200, 202, 203, 208, 210, 211, 213, 77	<code>\glsxtrp</code>
..... 198, 200, 202, 203, 208, 210, 211, 213, 76, 78	

<code>\glxtrparen</code>	158, 176–179, 181, 183, 184, 186, 188, 191–196, 198– 213, 215, 217–228, 230, 232–234, 236– 238, 248, 249, 251, 253, 256–258, 262, 263	<code>\glxtrshorthyphenlong</code>	257
<code>\Glsxtrpl</code>	43	<code>\glxtrshortlongdescname</code> 179, 194, 207, 222, 224, 246, 257, 260
<code>\glxtrpl</code>	43	<code>\glxtrshortlongdescsort</code> 179, 194, 207, 222, 224, 246, 257, 260
<code>\glxtrpostdescription</code>	134, 149, 289	<code>\GLSxtrshortpl</code>	15, 16, 267, 268
<code>\glxtrposthyphenlong</code>	259, 261	<code>\Glsxtrshortpl</code>	15, 268
<code>\glxtrposthyphenshort</code>	254, 255	<code>\glxtrshortpl</code>	15, 267
<code>\glxtrposthyphensubsequent</code> 254, 255, 259, 261	<code>\glxtrsmfont</code>	204
<code>\glxtrpostlink</code>	150	<code>\glxtrsmsuffix</code> 205, 206, 208, 209, 211, 212, 214, 216
<code>\glxtrpostlinkendsentence</code>	150	<code>\Glsxtrsubsequentfmt</code> 170, 173, 188, 198, 199, 211, 212, 228, 229, 231, 233, 251, 254, 259
<code>\glxtrpostlinkhook</code>	150	<code>\glxtrsubsequentfmt</code> 170, 173, 188, 197, 199, 211, 212, 228, 229, 231, 233, 251, 254, 259
<code>\glxtrpostlocalreset</code>	81, 83, 91	<code>\Glsxtrsubsequentplfmt</code> 170, 173, 188, 198, 199, 211, 213, 228, 230, 231, 233, 251, 254, 259
<code>\glxtrpostlocalunset</code>	81, 83, 91	<code>\glxtrsubsequentplfmt</code> 170, 173, 188, 197, 199, 211, 212, 228, 229, 231, 233, 251, 254, 259
<code>\glxtrpostlongdescription</code>	28	<code>\glxtrsupplocationurl</code>	107, 108
<code>\glxtrpostnamehook</code>	139–141	<code>\glxtrtagfont</code>	148
<code>\GlsXtrPostNewAbbreviation</code> 157, 173, 174, 176–180, 182, 184–187, 189–194, 196, 197, 200, 202, 204, 206– 209, 211, 214, 216, 218–227, 229, 231, 233–236, 239, 240, 242, 244, 245, 247, 249, 250, 252, 253, 255, 257–259, 261–263	<code>\Glsxtrtitlefirst</code>	265, 266, 278
<code>\glxtrpostreset</code>	81, 83, 91	<code>\glxtrtitlefirst</code>	265, 266, 277, 278
<code>\glxtrpostunset</code>	81, 83, 91	<code>\Glsxtrtitlefirstplural</code>	265, 266, 278
<code>\glxtrprotectlinks</code>	70–72	<code>\glxtrtitlefirstplural</code>	265, 266, 278
<code>\GlsXtrRecordCounter</code>	9	<code>\Glsxtrtitlefull</code>	265, 266, 280
<code>\glxtrregularfont</code>	47, 48, 53	<code>\glxtrtitlefull</code>	265, 266, 280
<code>\glxtrresourcecount</code>	113	<code>\Glsxtrtitlefullpl</code>	265, 266, 281
<code>\glxtrresourcefile</code>	113	<code>\glxtrtitlefullpl</code>	265, 266, 280
<code>\glxtrrestoremarkhook</code>	264	<code>\Glsxtrtitlefullpl</code>	265, 266, 279
<code>\glxtrscfont</code>	190	<code>\glxtrtitlelong</code>	265, 266, 279
<code>\glxtrscsuffix</code> 191, 193, 194, 196, 197, 199, 201, 203	<code>\glxtrtitlelong</code>	265, 266, 279
<code>\GlsXtrSetActualChar</code>	145	<code>\Glsxtrtitlelongpl</code>	265, 266, 279, 280
<code>\glxtrsetaliasnoindex</code>	11, 12, 68	<code>\glxtrtitlelongpl</code>	265, 266, 279
<code>\GlsXtrSetEncapChar</code>	146	<code>\Glsxtrtitleplural</code>	265, 266, 277
<code>\GlsXtrSetEscChar</code>	145	<code>\glxtrtitleplural</code>	265, 266, 277
<code>\glxtrsetfieldifexists</code>	25, 26	<code>\Glsxtrtitleshort</code>	265, 266, 276
<code>\GlsXtrSetLevelChar</code>	145	<code>\glxtrtitleshort</code>	265, 266, 275
<code>\glxtrsetpopts</code>	76	<code>\Glsxtrtitleshortpl</code>	265, 266, 276
<code>\glxtrsetupfulldefs</code> 160–163, 182, 203, 216, 237	<code>\glxtrtitleshortpl</code>	265, 266, 275, 276
<code>\GLSxtrshort</code>	15, 16, 80, 267, 268	<code>\Glsxtrtitletext</code>	265, 266, 277
<code>\Glsxtrshort</code>	15, 268	<code>\glxtrtitletext</code>	265, 266, 276
<code>\glxtrshort</code>	15, 267	<code>\glxtrtreetopindent</code>	290, 298
<code>\glxtrshortdescname</code> ...	185, 196, 209, 226	<code>\glxtrundefaction</code> ..	7, 11, 12, 21, 29, 31–33
<code>\glxtrshorthyphen</code>	260	<code>\glxtrundeftag</code>	21, 103, 104
		<code>\glxtrunsrtdo</code>	117

<code>\GlsXtrUseAbbrStyleFmts</code>	177, 179, 185, 187, 189, 190, 192, 194, 206, 208, 219, 221, 223, 224, 231, 234, 242, 244, 245, 247, 250–252, 256, 258, 261, 263	<code>\ifdefvoid</code> 33, 36–38, 71, 88, 104, 107, 118, 119
<code>\GlsXtrUseAbbrStyleSetup</code> 185, 187, 189, 190, 199, 212, 231, 234	<code>\ifdim</code>
<code>\glsextruserfield</code>	238	44, 45, 96, 291–298
<code>\glsextruserparen</code>	239–247	<code>\IfFileExists</code>
<code>\glsextrusersuffix</code>	239, 241, 243, 246	18, 108, 112, 113, 115, 283
<code>\glsextruseseealsoformat</code>	34, 35	<code>\ifglossaryexists</code>
<code>\glsextruseseeeformat</code>	34	33
<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	153, 174	<code>\ifglssacronym</code>
<code>\GlsXtrWarning</code>	41, 42	14, 111
<code>\glsextrword</code>	155	<code>\ifglssacrshortcuts</code>
<code>\glsextrwordsep</code> ..	155, 248, 250, 253, 256, 258	16
		<code>\ifglssautomake</code>
		100, 112, 115
		<code>\ifglssentrycounter</code>
		26
		<code>\ifglssentryexists</code>
		.. 31, 32, 41, 42, 45, 53, 131, 132, 149, 150
		<code>\ifglssfieldeq</code>
		130
		<code>\ifglsshfield</code>
		23, 68, 238
		<code>\ifglsshfieldlong</code>
		87, 88
		<code>\ifglsshfieldparent</code>
		118, 291–294
		<code>\ifglsshfieldshort</code>
		47, 48, 53
		<code>\ifglsshfieldsymbol</code>
		151, 290
		<code>\ifglssindexonlyfirst</code>
		69
		<code>\ifglssnogroupskip</code>
		285–289, 300
		<code>\ifglssnonumberlist</code>
		47
		<code>\ifglsssanitizesort</code>
		99
		<code>\ifglsssubentrycounter</code>
		26
		<code>\ifglssused</code>
		38, 66, 69, 84, 93, 96, 169, 291, 292, 294, 296, 297
		<code>\ifglssxindy</code>
		108, 110
		<code>\ifglssxtrinitwrglossbefore</code>
		50–52
		<code>\ifglssxtrininsertinside</code>
	 163–169, 171, 172, 176, 178, 180, 181, 183–189, 191–193, 195–205, 207– 220, 222–238, 240, 241, 243, 244, 246, 248, 250, 253, 255, 256, 258, 260, 262, 263
		<code>\ifHy@hyperindex</code>
		142
		<code>\ifinlistcs</code>
		24, 39
		<code>\ifKV@glsslink@hyper</code>
		48, 51, 52
		<code>\ifKV@glsslink@local</code>
		49
		<code>\ifKV@glsslink@noindex</code>
		8, 10, 68
		<code>\ifnum</code>
		12, 13, 84, 92, 93, 104, 113, 299
		<code>\ifstrequal</code>
		18
		<code>\ifthenelse</code>
		111, 112
		<code>\IfTrackedLanguageFileExists</code>
		281
		<code>\ifundef</code>
		72, 97, 147–149
		<code>\ifx</code>
		8, 9, 44, 45, 100, 106, 107, 115, 142–144, 146, 152, 154, 155, 157, 247, 300
		<code>\immediate</code>
		84, 92, 108, 109, 115
		<code>\index</code>
		143
		<code>\indexspace</code>
		300
		<code>\input</code>
		281
		<code>\inputencodingname</code>
		114
		<code>\istfilename</code>
		97
		<code>\item</code>
		110, 111, 284

H

<code>\hangindent</code>	290, 298
<code>\hbox</code>	284
<code>\hfill</code>	284
<code>\href</code>	71
<code>\hsize</code>	44
<code>\hss</code>	284
<code>\hyperlink</code>	71
<code>\hyperpage</code>	142
<code>\hyperref</code>	70, 108
<code>hyperref package</code>	72, 142, 264, 275

I

<code>\if</code>	40
<code>\if@glsextr@autoseeindex</code>	19, 20, 33, 36
<code>\if@glsextr@format@override</code>	142
<code>\if@glsextr@docdef@restricted</code>	39
<code>\if@glsextr@index@crossrefs</code>	13, 38
<code>\ifblank</code>	22, 41, 42, 97
<code>\ifcase</code>	7, 11, 17, 18, 39, 50, 101
<code>\ifcsdef</code> 7, 21, 27–31, 65, 76–80, 88, 100, 105, 118, 136–141, 150, 153, 169, 173, 284–288	
<code>\ifcsstring</code>	21, 132, 172
<code>\ifcsundef</code>	27, 29– 31, 39, 43, 46, 72, 83, 88–92, 104, 105, 108, 117, 132, 172, 174, 175, 283, 291, 298
<code>\ifcsvoid</code>	37, 131
<code>\ifdef</code>	12, 16, 20, 23, 25, 32, 35, 44, 67, 71, 77–79, 100, 102, 103, 114, 134, 135, 145, 146, 149, 238, 275–281, 284, 289
<code>\ifdefempty</code> .	7, 8, 25, 29–31, 33, 34, 51, 52, 82, 94, 97, 100, 107, 117, 119, 147, 155, 169
<code>\ifdefequal</code>	112, 119
<code>\ifdefstring</code>	6, 142, 148

J	
<code>\jobname</code>	108, 110–113, 115
K	
<code>\key@ifundefined</code> .	10, 11, 21, 22, 64, 116, 118
<code>\KV@glslink@hyperfalse</code>	53, 66, 67, 72
<code>\KV@glslink@noindexfalse</code>	67
<code>\KV@glslink@noindextrue</code>	67, 72
L	
<code>\LaTeX</code>	110, 111
<code>\leaders</code>	284
<code>\leavevmode</code>	28, 51
<code>\let</code>	5, 7–9, 11, 12, 14–17, 19–21, 23, 27, 28, 39, 40, 43–46, 48–70, 72, 73, 76, 82, 83, 91–105, 113–116, 119, 136– 144, 147–149, 152–156, 160–169, 171– 173, 182, 203, 216, 237, 264–266, 289, 291
<code>\letabbreviationstyle</code> ..	181, 183, 185, 187, 189, 190, 195, 197, 209, 210, 226, 227
<code>\letcs</code>	22, 25, 33, 34, 38, 64, 102–105, 118, 136–141
<code>\levelchar</code>	145
<code>\listadd</code>	88
<code>\listbreak</code>	148
<code>\listcsadd</code>	24
<code>\listcseadd</code>	24, 89
<code>\listcsgadd</code>	24, 39
<code>\listcsxadd</code>	24, 89
<code>\loadglsentries</code>	39, 109
<code>\long</code>	28
M	
<code>\MakeAcronymsAbbreviations</code>	96
<code>\makeatletter</code>	108, 113, 144
<code>\makeatother</code>	144
<code>\makebox</code>	284, 299
<code>\makefirstuc</code>	148
<code>makeglossaries</code>	102
<code>\makeglossaries</code>	97, 109–112, 115
<code>\makeglossary</code>	97
<code>makeindex</code>	301
<code>makeindex</code>	96
<code>\makenoidxglossaries</code>	110
<code>\MakeTextUppercase</code>	265
<code>\MakeUppercase</code>	265, 266
<code>\markboth</code>	265
<code>\markright</code>	265
<code>\maxdimen</code>	44, 45
<code>\mbox</code>	299
<code>\medskip</code>	111, 112, 117
<code>\MessageBreak</code>	39, 43, 84, 93, 99, 100, 172
<code>mfistuc</code> package	147, 148
<code>\mfistucMakeUppercase</code>	53– 64, 66, 74, 75, 77, 80, 87, 95, 120–130, 138, 140, 161, 163, 164, 166, 167, 169–171
<code>\mfu@checkword@arg</code>	148
<code>\mfu@checkword@do</code>	148
N	
<code>\NeedsTeXFormat</code>	5, 283
<code>\new@glossaryentry</code>	40, 99
<code>\new@ifnextchar</code> .	65, 66, 86, 87, 152, 160–169
<code>\newabbr</code>	15, 16
<code>\newabbreviation</code>	15, 16
<code>\newabbreviationhook</code>	157
<code>\newabbreviationstyle</code>	175, 177, 179, 180, 182, 183, 185, 187, 189–194, 196, 197, 199, 200, 202, 204–210, 212, 214, 215, 218–224, 226, 227, 229, 231, 232, 234, 236, 239, 240, 242, 244–246, 248–250, 252, 253, 255–257, 259–261, 263
<code>\newacronym</code>	94, 95
<code>\newacronymhook</code>	94
<code>\newacronymstyle</code>	95, 96
<code>\newcommand</code>	5–7, 9–34, 36, 38–43, 45–48, 50, 52, 53, 64–72, 75– 84, 86–96, 100–136, 141–155, 157–169, 171–175, 177, 179, 180, 185, 190, 204, 217, 238, 239, 248, 250, 252, 253, 256, 258, 261, 263, 265–281, 283, 289–291, 298
<code>\newcount</code>	12, 113
<code>\newentry</code>	16
<code>\newglossary</code>	14, 97
<code>\newglossaryentry</code> 16, 39, 40, 82, 90, 94, 134, 135, 157
<code>\newglossaryentry options</code>	
<code>alias</code>	13, 33, 35–38
<code>desc</code>	123, 124, 128, 129
<code>descplural</code>	124, 129
<code>first</code>	70, 121, 122, 127, 128, 175, 270, 271, 277, 301
<code>firstplural</code>	122, 128, 175, 271, 278, 301
<code>loclist</code>	24
<code>long</code>	126, 130, 278
<code>longplural</code>	126, 130, 279
<code>name</code>	119, 120, 127, 142
<code>plural</code>	121, 127, 175, 269, 270, 277
<code>see</code>	13, 20, 33, 35, 38, 40, 97

seealso	13, 33–35, 37, 38, 312	\org@ifKV@glslink@hyper	51, 52
short	125, 129, 154		
shortplural	126, 129, 154		
symbol	122, 123, 128		
symbolplural	123, 128		
text	70, 120, 127, 175, 177, 268, 269, 276		
\newif	50, 141, 175		
\newlength	290		
\newnum	16		
\newrobustcmd	23, 25, 26, 34, 35, 65, 66, 76, 77, 86, 87, 104, 118, 147, 148, 160–169, 247, 265, 267–275, 291–297		
\newsym	16		
\newterm	134		
\newtoks	154		
\newwrite	97		
\NoCaseChange	77–80, 267–275		
\noexpand	9, 10, 18, 34, 36, 37, 94, 108, 109, 113, 117, 143, 144, 157, 283		
\nofiles	111		
\noindent	111, 112		
\nopostdesc	28, 41, 42, 101, 134		
\nr	7, 11, 13, 16–18, 50, 101		
\ns@GLSxtrfull	161		
\ns@GLSxtrfull	160		
\ns@glxtrfull	160		
\ns@GLSxtrfullpl	162		
\ns@GLSxtrfullpl	162		
\ns@glxtrfullpl	161		
\ns@GLSxtrlong	165		
\ns@GLSxtrlong	165		
\ns@glxtrlong	164		
\ns@GLSxtrlongpl	169		
\ns@GLSxtrlongpl	168		
\ns@glxtrlongpl	168		
\ns@GLSxtrshort	164		
\ns@GLSxtrshort	163		
\ns@glxtrshort	163		
\ns@GLSxtrshortpl	167		
\ns@GLSxtrshortpl	166, 167		
\ns@glxtrshortpl	166		
\null	18		
\number	89–92, 113		
\numexpr	89, 92		
O			
\or	7, 11, 12, 17, 39, 40, 50		
\org@glossaryentrynumbers	45, 101		
\org@glossarytitle	100		
P			
\p@glS@hyp@opt	70		
package options:			
abbreviations	14		
accsupp	18, 119		
acronym	14		
automake	100, 110, 115		
true	115		
autoseeindex	20		
false	19		
debug			
showtargets	71		
docdef	12, 39, 82, 90		
false	39		
restricted	13		
true	40		
nonumberlist	45		
nopostdot			
false	14		
numbers	16		
record	7, 11, 48, 112, 310		
alsoindex	9		
shortcuts	16		
ac	16		
all	16		
false	16		
none	16		
true	16		
style	19		
stylemods	19		
symbols	16, 134		
undefaction	31, 32		
error	6		
warn	6		
xindy	35		
\PackageError	6, 9, 18, 20, 39, 43, 65, 66, 68, 76, 77, 82, 83, 90, 92, 93, 95, 97, 99, 105, 115, 118, 172–175, 283		
\PackageWarning	14		
\PackageWarningNoLine	14		
\pageref	26		
\par	111, 112, 289, 290, 299		
\parindent	290, 299		
\PassOptionsToPackage	5		
\preglossarypreamble	27		
\preto	67		
\print@noop@unsrtglossaryunit	10, 11		

<code>\theindex</code>	142	<code>\warn@noprintglossary</code>	98, 101
<code>\this@dialect</code>	281	<code>\write</code>	35, 84, 92, 97, 108, 109, 115
<code>\toks@</code>	107, 146		
<code>tracklang</code> package	114		
		X	
		<code>\x</code>	107
		<code>\xcapitalisewords</code>	136
U		<code>\xdef</code>	101, 117
<code>\undef</code>	11, 12, 147	<code>\xifinlist</code>	88
<code>\underline</code>	148	<code>\xifinlistcs</code>	24
<code>\unskip</code>	28, 39, 284	<code>xindy</code>	301
<code>\usepackage</code>	111, 112	<code>xindy</code>	96
		<code>xkeyval</code> package	5
V		<code>\XKV@checkchoice</code>	47
<code>\val</code>	7, 11, 13, 16–18, 50, 101	<code>\XKV@plfalse</code>	47
		<code>\XKV@resa</code>	47
W		<code>\XKV@sttrue</code>	47
<code>\warn@nomakeglossaries</code>	98		