

glossaries-extra.sty v1.48: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2021-11-22

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	34
1.3	Modifications to Commands Provided by glossaries	52
1.3.1	Existence Checks	57
1.3.2	Document Definitions	68
1.3.3	Existing Glossary Style Modifications	74
1.3.4	Entry Formatting, Hyperlinks and Indexing	78
1.3.5	Entry Counting	119
1.3.6	Acronym Modifications	135
1.3.7	Indexing and Displaying Glossaries	139
1.4	Link Counting	182
1.5	Integration with glossaries-accsupp	184
1.6	Categories	201
1.7	Abbreviations	230
1.7.1	Abbreviation Styles Setup	250
1.7.2	Predefined Styles (Default Font)	254
1.7.3	Predefined Styles (Small Capitals)	273
1.7.4	Predefined Styles (Fake Small Capitals)	290
1.7.5	Predefined Styles (Emphasized)	307
1.7.6	Predefined Styles (User Parentheses Hook)	331
1.7.7	Predefined Styles (Hyphen)	345
1.7.8	Predefined Styles (No Short on First Use)	360
1.8	Using Entries in Headings	365
1.9	Multi (Combined/Compound) Entries	388
1.10	Multi-Lingual Support	438
1.11	glossaries-extra-bib2gls.sty	439
2	Style Adjustments (glossaries-extra-stylemods.sty)	488
2.1	Package Initialisation	488
2.2	List-Like Styles	489
2.3	Longtable Styles	493
2.4	Long Ragged Styles	495
2.5	Supertabular Styles	497
2.6	Super Ragged Styles	499
2.7	Inline Style	501
2.8	Tree Styles	501

2.9 Multicolumn Styles	521
3 bookindex style (glossary-bookindex.sty)	528
3.1 Package Initialisation and Options	528
4 longextra styles (glossary-longextra.sty)	535
4.1 Package Initialisation and Options	535
5 topic styles (glossary-topic.sty)	558
5.1 Package Initialisation and Options	558
Glossary	564
Change History	565
Index	595

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2021/11/22 v1.48 (NLCT)]
```

Requires xkeyval to define package options.

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

Requires etoolbox package.

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

Has glossaries already been loaded?

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

```
6 {%
```

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

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

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

```
9 }
```

```
10 {%
```

Not already loaded, so pass options to glossaries.

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

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

```
13 }%
```

Set the defaults.

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

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

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

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

```
18 {}%
```

```
19 {%
```

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

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

```
22 {}%
```

```
23 }%
```

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

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

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

```
27 }
```

```
28 }
```

Declare package options.

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

`\f@for@gl@sentries`

```

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

```

`\f@for@gl@sentries`

```

48 \newcommand*{\@glstr@do@redef@for@gl@sentries}{%
49   \renewcommand*{\for@gl@sentries}[3][\gl@defaulttype]{%
50     \protected@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 }%

```

undefaction

```

63 \define@choicekey{glossaries-extra.sty}{undefaction}%
64 [\glstr@undefaction@val\glstr@undefaction@nr]%
65 {warn,error}%
66 {%
67   \ifcase\glstr@undefaction@nr\relax
68     \let\glstrundefaction\glstr@warn@undefaction
69     \let\glstr@warnonexistssordo\glstr@warn@onexistssordo
70     \let\glstr@redef@for\glstr@do@redef@for\glstr@sentries
71   \or
72     \let\glstrundefaction\glstr@err@undefaction
73     \let\glstr@warnonexistssordo\glstr@gobble
74     \let\glstr@redef@for\glstr@redef@for\glstr@sentries\relax
75   \fi
76 }

```

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

\glstr@record Does nothing by default.

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

```

\glstr@recordsee Does nothing by default.

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

```

\glstr@defaultnumberformat

```
79 \newcommand*{\glstr@defaultnumberformat}{\glstr@numberformat}%

```

\glstr@setDefaultNumberFormat

```

80 \newcommand*{\glstr@setDefaultNumberFormat}[1]{%
81   \renewcommand*{\glstr@defaultnumberformat}{#1}%
82 }%

```

The record option is somewhat problematic. On the first \LaTeX run the entries aren't defined. This isn't as straight-forward as commands like `\cite` since attributes associated with the entry's category may switch off the indexing or the entry's glossary type might require a particular counter. This kind of information can't be determined until the entry has been defined. So there are two different commands here. One that's used if the entry hasn't been defined, which tries to use sensible defaults, and one which is used when the entry has been defined.

cord@wrglossary The record=only option sets \@do@wrglossary to this command, which means it's done within \glsadd and \@gls@link, and so is only done if the entry exists.

```

83 \newcommand*{\@glsxtr@do@record@wrglossary}[1]{%
84   \begingroup
85     \ifKV@glslink@noindex
86     \else

87     \protected@edef\@gls@label{\glsdetoklabel{#1}}%
88     \let\glslabel\@gls@label
89     \glswriteentry{#1}%
90     {%
91       \ifdefempty{\@glsxtr@thevalue}%
92       {%
93         \ifx\@glsxtr@org@theHvalue\@glsxtr@theHvalue
94         \else
95           \let\theHglsentrycounter\@glsxtr@theHvalue
96           \fi
97           \glsxtr@saveentrycounter
98           \let\@do@wrglossary\@glsxtr@dorecord
99       }%
100     {%
101       \let\theHglsentrycounter\@glsxtr@thevalue
102       \let\theHglsentrycounter\@glsxtr@theHvalue
103       \let\@do@wrglossary\@glsxtr@dorecordnodefer
104     }%
105     \ifx\@glsxtr@record@setting\@glsxtr@record@setting@alsoindex
106     \glsxtr@do@wrglossary{#1}%
107     \else
108     \@glsxtrwrglossmark

```

Increment associated counter.

```

109     \glsxtr@inc@wrglossaryctr{#1}%
110     \@do@wrglossary
111   \fi
112 }%
113 \fi
114 \endgroup
115 }

```

index@wrglossary The record=alsoindex option needs to both record and index.

```

116 \newcommand*{\@glsxtr@do@alsoindex@wrglossary}[1]{%
117   \glsxtr@do@wrglossary{#1}%
118   \@glsxtr@do@record@wrglossary{#1}%
119 }

```

@glsxtr@record The record=only option sets \@glsxtr@record to this. This performs the recording if the entry *doesn't exist* and is done at the start of \@gls@field@link and commands like \@gls@ (before the existence test). This means that it disregards the wrgloss key.

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 second argument is the entry's label. The third argument is the key family (`glslink` in most cases, `glossadd` for `\glsadd`).

```
120 \newcommand*{\@@glsxtr@record}[3]{%
```

Save the label in case it's needed. This needs to be outside the existence check to allow the post-link hook to reference it.

```
121 \protected@edef\@gls@label{\glsdetoklabel{#2}}%
122 \let\glslabel\@gls@label
123 \ifglsentryexists{#2}{}%
124 {%
125   \@@glsxtrwrglossmark
126   \beginngroup
127     \let\@glsnumberformat\@glsxtr@defaultnumberformat
128     \def\@glsxtr@thevalue{}%
129     \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
130     \let\@glsxtr@org@theHvalue\@glsxtr@theHvalue
```

Entry hasn't been defined, so we'll have to assume it's `\glscounter` by default.

```
131   \let\@gls@counter\glscounter
```

Unless the equations option is on and this is inside a numbered maths environment.

```
132   \ifglsxtr@equations
133     \@glsxtr@use@equation@counter
134   \fi
```

Check for default options (which may switch off indexing).

```
135   \@gls@setdefault@glslink@opts
```

Implement any pre-key settings.

```
136   \csuse{\@glsxtr@#3@prekeys}%
```

Assign keys.

```
137   \setkeys{#3}{#1}%
```

Implement any post-key settings. Is the auto-add on?

```
138   \glsxtr@do@autoadd{#3}%
```

Check post-key hook.

```
139   \csuse{\@glsxtr@#3@postkeys}%
```

Increment associated counter.

```
140   \glsxtr@inc@wrglossaryctr{#2}%
```

Check if `noindex` option has been used.

```
141   \ifKV@glslink@noindex
142   \else
143     \glswriteentry{#2}%
144   {%
```

Check if `thevalue` has been set.

```
145     \ifdefempty{\@glsxtr@thevalue}%
146     {%
```

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

```

147         \ifx\@glxtr@org@theHvalue\@glxtr@theHvalue
148         \else
149         \let\theHglentrycounter\@glxtr@theHvalue
150         \fi

```

Save the entry counter.

```

151         \glxtr@saveentrycounter

```

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

```

152         \let\@@do@@wrglossary\@glxtr@dorecord
153     }%
154     {%

```

thevalue 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.)

```

155         \let\theHglentrycounter\@glxtr@thevalue
156         \let\theHglentrycounter\@glxtr@theHvalue
157         \let\@@do@@wrglossary\@glxtr@dorecordnodefer
158     }%
159     \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
160     \glxtr@@do@@wrglossary{#2}%
161     \else

```

No need to escape special characters.

```

162         \@@do@@wrglossary
163     \fi
164 }%
165 \fi
166 \endgroup
167 }%
168 }

```

glslink@prekeys

```

169 \newcommand{\@glxtr@glslink@prekeys}{\glslinkpresetkeys}

```

lslink@postkeys

```

170 \newcommand{\@glxtr@glslink@postkeys}{\glslinkpostsetkeys}

```

lossadd@prekeys

```

171 \newcommand{\@glxtr@glossadd@prekeys}{\glsaddpresetkeys}

```

ossadd@postkeys

```

172 \newcommand{\@glxtr@glossadd@postkeys}{\glsaddpostsetkeys}

```

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

```

173 \newcommand*\@glxtr@dorecord{%

```

```

174 \global\let\@glsrecordlocref\theglentrycounter
175 \let\@glsxtr@orgprefix\@glo@counterprefix
176 \ifx\theglentrycounter\theHglentrycounter
177   \def\@glo@counterprefix{}%
178 \else
  Protect against non-expandable commands occurring in the location.
179   \protected@edef\@glsxtr@theentrycounter{\theglentrycounter}%
180   \protected@edef\@glsxtr@theHentrycounter{\theHglentrycounter}%
181   \@onelevel@sanitize\@glsxtr@theentrycounter
182   \@onelevel@sanitize\@glsxtr@theHentrycounter
183   \protected@edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
184     {\@glsxtr@theentrycounter}{\@glsxtr@theHentrycounter}%
185   }%
186   \@do@gls@getcounterprefix
187 \fi

```

Don't protect the \@glsrecordlocref from premature expansion. If the counter isn't

page then it needs expanding. If the location includes \thepage then \protected@write will automatically deal with it.

```

188 \ifx\@glsxtr@record@setting\@glsxtr@record@setting@nameref
189   \@glsxtr@do@nameref@record
190   {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
191   {\@glsrecordlocref}%
192 \else
193   \protected@write\@auxout{}{\string\@glsxtr@record
194     {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
195     {\@glsrecordlocref}}%
196 \fi
197 \@glsxtr@counterrecordhook
198 \let\@glo@counterprefix\@glsxtr@orgprefix
199 }

```

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

```

200 \newcommand*\@glsxtr@dorecordnodefer{%
201   \ifx\theglentrycounter\theHglentrycounter
202     \ifx\@glsxtr@record@setting\@glsxtr@record@setting@nameref
203       \@glsxtr@do@nameref@record
204       {\@gls@label}{\@gls@counter}{\@glsnumberformat}%
205       {\theglentrycounter}%
206     \else
207       \protected@write\@auxout{}{\string\@glsxtr@record
208         {\@gls@label}{\@gls@counter}{\@glsnumberformat}%
209         {\theglentrycounter}}%
210     \fi
211   \else
212     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix

```

```

213     {\theglentrycounter}{\theHglentrycounter}%
214 }%
215 \@do@gl@getcounterprefix
216 \ifx\@glxtr@record@setting\@glxtr@record@setting@nameref
217   \@glxtr@do@nameref@record
218     {\@gl@label}{\@glo@counterprefix}{\@gl@counter}%
219     {\@gl@numberformat}{\theglentrycounter}%
220 \else
221   \protected@write\@auxout{}\string\glxtr@record
222     {\@gl@label}{\@glo@counterprefix}{\@gl@counter}{\@gl@numberformat}%
223     {\theglentrycounter}}%
224 \fi
225 \fi
226 \@glxtr@counterrecordhook
227 }

```

`\glxtr@ifnum@mmode` Check if in a numbered maths environment. The `amsmath` package is automatically loaded by `datatool-base`, which is required by `glossaries`, so `\ifst@rred` and `\if@display` should both be defined.

```

228 \newcommand{\@glxtr@ifnum@mmode}[2]{%
229   \ifmmode
230     \ifst@rred
231       #2%
232   \else

```

Non-`amsmath` environments and regular inline math mode isn't flagged as starred by `amsmath`, but we can't use `\mathchoice` in this case as it's not the current style that's relevant. Instead we can use `amsmath's \if@display`. This may not work for environments that aren't provided by `amsmath`.

```

233     \if@display #1\else #2\fi
234   \fi
235 \else
236   #2%
237 \fi
238 }

```

`\glxtr@nameref@record` With `record=nameref`, the current label information is included in the record, but this may not have been defined, so `\csuse` will prevent an undefined control sequence error and just leave the last two arguments blank if there's no information. In the event that a record is in `amsmath's align` environment `\@currentHref` will be out. There may be other instances where `\@currentHref` is out, so this also saves `\theHglentrycounter`, which is useful if it can't be obtained by prefixing `\theglentrycounter`.

```

239 \newcommand*{\@glxtr@do@nameref@record}[5]{%
240   \gl@ifnotmeasuring
241   {%
242     \protected@write\@auxout{}\string\glxtr@record@nameref
243       {#1}{#2}{#3}{#4}{#5}%
244     {\csuse{@currentlabelname}}{\csuse{@currentHref}}}%

```

```

245     {\theHglentrycounter}}%
246   }%
247 }

r@recordcounter
248 \newcommand*{\@glxtr@recordcounter}{%
249   \@glxtr@noop@recordcounter
250 }

p@recordcounter
251 \newcommand*{\@glxtr@noop@recordcounter}[1]{%
252   \PackageError{glossaries-extra}{\string\GlsXtrRecordCounter\space
253     requires record=only or record=hybrid package option}{}%
254 }

p@recordcounter
255 \newcommand*{\@glxtr@op@recordcounter}[1]{%
256   \protected@eappto\@glxtr@counterrecordhook{\noexpand\@glxtr@docounterrecord{#1}}}%
257 }

lsxtr@recordsee Deal with \glssee in record mode. (This doesn't increment the associated counter.)
258 \newcommand*{\@glxtr@recordsee}[2]{%
259   \@glxtr@wrglossmark
260   \def\@gls@xref{#2}%
261   \@onelevel@sanitize\@gls@xref
262   \protected@write\@auxout{}\{\string\glxtr@recordsee{#1}-{\@gls@xref}}%
263 }

srtglossaryunit
264 \newcommand{\printunsrtglossaryunit}{%
265   \print@noop@unsrtglossaryunit
266 }

tr@setup@record Initialise.
267 \newcommand*{\glxtr@setup@record}{\let\@do@wrglossary\glxtr@do@wrglossary}

saveentrycounter Only store the entry counter information if the indexing is on.
268 \newcommand*{\glxtr@indexonly@saveentrycounter}{%
269   \ifKV@glslink@noindex
270   \else
271     \glxtr@saveentrycounter
272   \fi
273 }

addloclistfield
274 \newcommand*{\glxtr@addloclistfield}{%
275   \key@ifundefined{glossentry}{loclist}%
276   {%

```

```

277 \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
278 \appto\@gls@keymap{,{loclist}{loclist}}%
279 \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
280 \appto\@newglossaryentryposthook{%
281   \gls@assign@field{\@glo@label}{loclist}{\@glo@loclist}%
282 }%
283 \glssetnoexpandfield{loclist}%
284 }%
285 {}%

```

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

```

286 \key@ifundefined{glossentry}{location}%
287 {%
288   \define@key{glossentry}{location}{\def\@glo@location{##1}}%
289   \appto\@gls@keymap{,{location}{location}}%
290   \appto\@newglossaryentryprehook{\def\@glo@location{}}%
291   \appto\@newglossaryentryposthook{%
292     \gls@assign@field{\@glo@label}{location}{\@glo@location}%
293   }%
294   \glssetnoexpandfield{location}%
295 }%
296 {}%

```

Add a key to store the group heading.

```

297 \key@ifundefined{glossentry}{group}%
298 {%
299   \define@key{glossentry}{group}{\def\@glo@group{##1}}%
300   \appto\@gls@keymap{,{group}{group}}%
301   \appto\@newglossaryentryprehook{\def\@glo@group{}}%
302   \appto\@newglossaryentryposthook{%
303     \gls@assign@field{\@glo@label}{group}{\@glo@group}%
304   }%
305   \glssetnoexpandfield{group}%
306 }%
307 {}%
308 }

```

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

```

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

```

etting@alsoindex As from v1.46, the record=alsoindex is renamed to record=hybrid with record=alsoindex as a deprecated synonym to avoid confusion. The internal commands that include alsoindex in the name will remain unchanged to avoid breaking things, but this command will need to be redefined by record=hybrid.

```

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

```

rd@setting@only

```

311 \newcommand*{\@glsxtr@record@setting@only}{only}

```

setting@nameref

```
312 \newcommand*{\@glxtr@record@setting@nameref}{nameref}
```

@if@record@only

```
313 \newcommand*{\@glxtr@if@record@only}[2]{%
314   \ifx\@glxtr@record@setting\@glxtr@record@setting@only
315     #1%
316   \else
317     \ifx\@glxtr@record@setting\@glxtr@record@setting@nameref
318       #1%
319     \else
320       #2%
321     \fi
322   \fi
323 }
```

ord@setting@off

```
324 \newcommand*{\@glxtr@record@setting@off}{off}
```

id@noprintgloss Used by hybrid method if \printglossary isn't used.

```
325 \newcommand\@glxtr@warn@hybrid@noprintgloss{%
326   \ifdefstring{\@glo@types}{,}%
327   {%
328     \GlossariesExtraWarningNoLine{No glossaries have been defined}%
329   }%
330   {%
331     \GlossariesExtraWarningNoLine{No \string\printglossary\space
332       or \string\printglossaries\space
333       found. ^^JYou have requested the hybrid setting
334       record=\@glxtr@record@setting\space which requires a
335       combination of bib2gls (to fetch entries) and makeindex/xindy
336       (to sort and collate the entries). If you only want to use
337       bib2gls then change the option to record=only or record=nameref}%
338   }%
339 }
```

cord@only@setup Initialisation code for record=only and record=nameref

```
340 \newcommand*{\@glxtr@record@only@setup}{%
341   \def\glxtr@setup@record{%
342     \@glxtr@autoseeindexfalse
343     \let\@do@seeglossary\@glxtr@recordsee
344     \let\@glxtr@record\@glxtr@record
345     \let\@do@wrglossary\@glxtr@do@record@wrglossary
346     \let\@gls@saveentrycounter\relax
347     \let\glxtrundefaction\@glxtr@warn@undefaction
348     \let\glxtr@warnonexistsordo\@glxtr@warn@onexistsordo
349     \glxtr@addloclistfield
350     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
351     \let\@glxtr@recordcounter\@glxtr@op@recordcounter
```

```

352 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
Switch off the index suppression for aliased entries. (bib2gls will deal with them.)
353 \def\glstrsetaliasnoindex{}%
\@glstrsetupsort@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.
354 \ifdef\@glstrsetupsort@none{\@glstrsetupsort@none}{}%
Warn about using \printglossary:
355 \def\glstrNoGlossaryWarning{\@glstr@record@noglossarywarning}%
Load glossaries-extra-bib2gls:
356 \RequirePackage{glossaries-extra-bib2gls}%
357 }%
358 }

```

record Now define the record package option. As from v1.46, record=alsoindex is a deprecated synonym of record=hybrid to avoid confusion.

```

359 \define@choicekey{glossaries-extra.sty}{record}
360 [ \@glstr@record@setting\glstr@record@nr ]%
361 {off,only,alsoindex,nameref,hybrid}%
362 [only]%
363 {%
364 \ifcase\glstr@record@nr\relax
Don't record.
365 \def\glstr@setup@record{%
366 \renewcommand*{\@do@seeglossary}{\@glstr@doseeglossary}%
367 \renewcommand*{\@glstr@record}[3]{}%
368 \let\@do@wrglossary\glstr@do@wrglossary
369 \let\@glstr@saveentrycounter\glstr@indexonly@saveentrycounter
370 \let\glstrundefaction\@glstr@err@undefaction
371 \let\glstr@warnonexistsordo\@gobble
372 \let\@glstr@recordcounter\@glstr@noop@recordcounter
373 \def\printunsrtglossaryunit{\print@noop@unsrtglossaryunit}%
374 \undef\glstrsetaliasnoindex
375 }%
376 \or

```

Only record (don't index).

```

377 \@glstr@record@only@setup
378 \or

```

Record and index. This option doesn't load glossaries-extra-bib2gls as the sorting is performed by xindy or makeindex. Index in this sense refers to the indexing mechanism used with indexing applications such as makeindex and xindy, but this could be confused with recording locations so "alsoindex" is now deprecated in favour of "hybrid", which is more obvious.

```

379 \def\glstr@setup@record{%
380 \renewcommand*{\@glstr@record@setting@alsoindex}{alsoindex}%

```



```

381 \renewcommand*{\@do@seeglossary}{\@glxtr@dosee@alsoindex@glossary}%
382 \let\@glxtr@record\@glxtr@record
383 \let\@do@wrglossary\glxtr@do@alsoindex@wrglossary
384 \let\@glxtr@saveentrycounter\glxtr@indexonly@saveentrycounter
385 \let\glxtrundefaction\glxtr@warn@undefaction
386 \let\glxtr@warnonexistsordo\glxtr@warn@onexistsordo
387 \glxtr@addloclistfield
388 \let\@glxtr@recordcounter\glxtr@op@recordcounter
389 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
390 \undef\glxtrsetaliasnoindex
391 }%
392 \or

```

Only record (don't index) but also include nameref information.

```

393 \@glxtr@record@only@setup
394 \ifundef\hyperlink
395 {\GlossariesExtraWarning{You have requested record=nameref but
396 the document doesn't support hyperlinks}}%
397 }%
398 \or

```

Hybrid record (use bib2gls to fetch definitions) and index (use makeindex/xindy to sort and collate).

```

399 \def\glxtr@setup@record{%
400 \renewcommand*{\@glxtr@record@setting@alsoindex}{hybrid}%
401 \renewcommand*{\@do@seeglossary}{\@glxtr@dosee@alsoindex@glossary}%
402 \let\@glxtr@record\@glxtr@record
403 \let\@do@wrglossary\glxtr@do@alsoindex@wrglossary
404 \let\@glxtr@saveentrycounter\glxtr@indexonly@saveentrycounter
405 \let\glxtrundefaction\glxtr@warn@undefaction
406 \let\glxtr@warnonexistsordo\glxtr@warn@onexistsordo
407 \glxtr@addloclistfield
408 \let\@glxtr@recordcounter\glxtr@op@recordcounter
409 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
410 \undef\glxtrsetaliasnoindex
411 }%
412 \fi
413 }

```

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).

```

414 \newcommand*{\@glxtr@docdefval}{0}

```

Need to provide conditional commands that are backward compatible:

`\if@glxtrdocdef`

```

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

```

```

lsxtrdocdeftrue
416 \newcommand*{\@glxtrdocdeftrue}{\def\@glxtr@docdefval{1}}

sxtrdocdeffalse
417 \newcommand*{\@glxtrdocdeffalse}{\def\@glxtr@docdefval{0}}

docdef 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.
418 \define@choicekey{glossaries-extra.sty}{docdef}
419 [\@glxtr@docdefsetting\@glxtr@docdefval]%
420 {false,true,restricted,atom}[true]%
421 {%
422 \ifnum\@glxtr@docdefval>1\relax
423 \renewcommand*{\@glsdofexistsorwarn}{\glsdofexists}%
424 \else
425 \renewcommand*{\@glsdofexistsorwarn}{\glsdofexistsorwarn}%
426 \fi
427 }

ocdefrestricted
428 \newcommand*{\ifglxtrdocdefrestricted}{\ifnum\@glxtr@docdefval>1 }

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

indexcrossrefs Automatically index cross references at the end of the document
430 \define@boolkey{glossaries-extra.sty}[\@glxtr]{indexcrossrefs}[true]{%
431 \ifglxtrindexcrossrefs
432 \else
433 \renewcommand*{\@glxtr@autoindexcrossrefs}{}}%
434 \fi
435 }

Switch off since this can increase the build time.
436 \@glxtrindexcrossrefsfalse

But allow see key to switch it on automatically.

oindexcrossrefs
437 \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.
438 \define@boolkey{glossaries-extra.sty}[\@glxtr@]{autoseeindex}[true]{%
439 }
440 \@glxtr@autoseeindextrue

```

equations Provide a boolean option to automatically switch to the equation counter when in a numbered maths environment.

```

441 \define@boolkey{glossaries-extra.sty}[@glsxtr@]{equations}[true]{%
442 }
443 \@glsxtr@equationsfalse

\glsxtr@float
444 \let\glsxtr@float\@float

glsxtr@dblfloat
445 \let\glsxtr@dblfloat\@dblfloat

floats Provide a boolean option to automatically switch to the the corresponding counter when in a float.
446 \define@boolkey{glossaries-extra.sty}[@glsxtr@]{floats}[true]{%
447   \if@glsxtr@floats
448     \renewcommand*{\@float}[1]{\renewcommand{\glscounter}{##1}\glsxtr@float{##1}}%
449     \renewcommand*{\@dblfloat}[1]{\renewcommand{\glscounter}{##1}\glsxtr@dblfloat{##1}}%
450   \else
451     \let\@float\glsxtr@float
452     \let\@dblfloat\glsxtr@dblfloat
453   \fi
454 }
455 \@glsxtr@floatsfalse

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

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

459 \@glsxtr@declareoption{nowarn}{%
460   \let\GlossariesExtraWarning\@gobble
461   \let\GlossariesExtraWarningNoLine\@gobble
462   \glsxtr@doption{nowarn}%
463 }

xtr@defpostpunc Redefines \glspostdescription. The postdot and nopostdot options will have to redefine this.
464 \newcommand*{\@glsxtr@defpostpunc}{%

postdot Shortcut for nopostdot=false
465 \@glsxtr@declareoption{postdot}{%
466   \glsxtr@doption{nopostdot=false}%
467   \renewcommand*{\@glsxtr@defpostpunc}{%
468     \renewcommand*{\glspostdescription}{%
469       \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%

```

```

470 }%
471 }

```

nopostdot Needs to redefine \@glsxtr@defpostpunc

```

472 \define@choicekey{glossaries-extra.sty}{nopostdot}{true,false}[true]{%
473   \glsxtr@doption{nopostdot=#1}%
474   \renewcommand*{\@glsxtr@defpostpunc}{%
475     \renewcommand*{\glspostdescription}{%
476       \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
477   }%
478 }

```

postpunc Set the post-description punctuation. This also sets the \ifglsnopostdot conditional, which now indicates if the post-description punctuation has been suppressed.

```

479 \define@key{glossaries-extra.sty}{postpunc}{%
480   \glsxtr@doption{nopostdot=false}%
481   \ifstrequal{#1}{dot}%
482   {%
483     \renewcommand*{\@glsxtr@defpostpunc}{%
484       \renewcommand*{\glspostdescription}{.\spacefactor\sfcode'\. }%
485     }%
486   }%
487   {%
488     \ifstrequal{#1}{comma}%
489     {%
490       \renewcommand*{\@glsxtr@defpostpunc}{%
491         \renewcommand*{\glspostdescription}{,}%
492       }%
493     }%
494     {%
495       \ifstrequal{#1}{none}%
496       {%
497         \glsxtr@doption{nopostdot=true}%
498         \renewcommand*{\@glsxtr@defpostpunc}{%
499           \renewcommand*{\glspostdescription}{}%
500         }%
501       }%
502       {%
503         \renewcommand*{\@glsxtr@defpostpunc}{%
504           \renewcommand*{\glspostdescription}{#1}%
505         }%
506       }%
507     }%
508   }%
509 }

```

glsxtrabbrvtype Glossary type for abbreviations.

```

510 \newcommand*{\glsxtrabbrvtype}{\glsdefaulttype}

```

bbreviationsdef Set by abbreviations option.

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

bbreviationsdef

```
512 \newcommand*{\@glxtr@doabbreviationsdef}{%
513   \ifpackageloaded{babel}%
514   {\providecommand{\abbreviationsname}{\acronymname}}%
515   {\providecommand{\abbreviationsname}{Abbreviations}}%
516   \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
517   \renewcommand*{\glxtrabbrvtype}{abbreviations}%
518   \newcommand*{\printabbreviations}[1][1]{%
519     \printglossary[type=\glxtrabbrvtype,##1]%
520   }%
521   \disable@keys{glossaries-extra.sty}{abbreviations}%
```

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

```
522   \ifglssacronym
523   \else
524     \renewcommand*{\acronymtype}{\glxtrabbrvtype}%
525   \fi
526 }%
```

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

```
527 \@glxtr@declareoption{abbreviations}{%
528   \let\@glxtr@abbreviationsdef\@glxtr@doabbreviationsdef
529 }
```

iationShortcuts Enable shortcut commands for the abbreviations. Unlike the analogous command provided by glossaries, this uses \newcommand instead of \let as a safety feature (except for \newabbr which is also provided with \GlsXtrDefineAcShortcuts).

```
530 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
531   \newcommand*{\ab}{\cglsl}%
532   \newcommand*{\abp}{\cglspl}%
533   \newcommand*{\as}{\glxtrshort}%
534   \newcommand*{\asp}{\glxtrshortpl}%
535   \newcommand*{\al}{\glxtrlong}%
536   \newcommand*{\alp}{\glxtrlongpl}%
537   \newcommand*{\af}{\glxtrfull}%
538   \newcommand*{\afp}{\glxtrfullpl}%
539   \newcommand*{\Ab}{\cGls}%
540   \newcommand*{\Abp}{\cGlspl}%
541   \newcommand*{\As}{\Glsxtrshort}%
542   \newcommand*{\Asp}{\Glsxtrshortpl}%
543   \newcommand*{\Al}{\Glsxtrlong}%
544   \newcommand*{\Alp}{\Glsxtrlongpl}%
545   \newcommand*{\Af}{\Glsxtrfull}%
546   \newcommand*{\Afp}{\Glsxtrfullpl}%
547   \newcommand*{\AB}{\cGLS}%
548   \newcommand*{\ABP}{\cGLSpl}%
```

```

549 \newcommand*{\AS}{\GLSxtrshort}%
550 \newcommand*{\ASP}{\GLSxtrshortpl}%
551 \newcommand*{\AL}{\GLSxtrlong}%
552 \newcommand*{\ALP}{\GLSxtrlongpl}%
553 \newcommand*{\AF}{\GLSxtrfull}%
554 \newcommand*{\AFP}{\GLSxtrfullpl}%

555 \providecommand*{\newabbr}{\newabbreviation}%

  Disable this command after it's been used.
556 \let\GlsXtrDefineAbbreviationShortcuts\relax
557 }

```

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

```

558 \newcommand*{\GlsXtrDefineAcShortcuts}{%
559   \newcommand*{\ac}{\cglS}%
560   \newcommand*{\acp}{\cglSpl}%
561   \newcommand*{\acs}{\glSxtrshort}%
562   \newcommand*{\acsp}{\glSxtrshortpl}%
563   \newcommand*{\acl}{\glSxtrlong}%
564   \newcommand*{\aclp}{\glSxtrlongpl}%
565   \newcommand*{\acf}{\glSxtrfull}%
566   \newcommand*{\acfp}{\glSxtrfullpl}%
567   \newcommand*{\Ac}{\cGls}%
568   \newcommand*{\Acp}{\cGlspl}%
569   \newcommand*{\Acs}{\Glsxtrshort}%
570   \newcommand*{\Acsp}{\Glsxtrshortpl}%
571   \newcommand*{\Acl}{\Glsxtrlong}%
572   \newcommand*{\Aclp}{\Glsxtrlongpl}%
573   \newcommand*{\Acf}{\Glsxtrfull}%
574   \newcommand*{\Acfp}{\Glsxtrfullpl}%
575   \newcommand*{\AC}{\cGLS}%
576   \newcommand*{\ACP}{\cGLSpl}%
577   \newcommand*{\ACS}{\GLSxtrshort}%
578   \newcommand*{\ACSP}{\GLSxtrshortpl}%
579   \newcommand*{\ACL}{\GLSxtrlong}%
580   \newcommand*{\ACLP}{\GLSxtrlongpl}%
581   \newcommand*{\ACF}{\GLSxtrfull}%
582   \newcommand*{\ACFP}{\GLSxtrfullpl}%

583   \providecommand*{\newabbr}{\newabbreviation}%

  Disable this command after it's been used.
584   \let\GlsXtrDefineAcShortcuts\relax
585 }

```

`\defineOtherShortcuts` Similarly provide shortcut versions for the commands provided by the symbols and numbers options.

```

586 \newcommand*{\GlsXtrDefineOtherShortcuts}{%

```

```

587 \newcommand*{\newentry}{\newglossaryentry}%
588 \ifdef\printsymbols
589 {%
590   \newcommand*{\newsym}{\glstrnewsymbol}%
591 }{}%
592 \ifdef\printnumbers
593 {%
594   \newcommand*{\newnum}{\glstrnewnumber}%
595 }{}%
596 \let\GlsXtrDefineOtherShortcuts\relax
597 }

```

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.

```
598 \newcommand*{\@glstr@setupshortcuts}{}%
```

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

```
599 \newcommand*{\@glstr@shortcutsval}{\ifglscrshortcuts acro\else none\fi}%

```

shortcuts 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).

```

600 \define@choicekey{glossaries-extra.sty}{shortcuts}%
601 [\@glstr@shortcutsval\@glstr@shortcutsnr]%
602 {acronyms,acro,abbreviations,abbr,other,all,true,ac,none,false}[true]{%
603   \ifcase\@glstr@shortcutsnr\relax % acronyms
604     \renewcommand*{\@glstr@setupshortcuts}{%
605       \glscrshortcuttrue
606       \DefineAcronymSynonyms
607     }%
608   \or % acro
609     \renewcommand*{\@glstr@setupshortcuts}{%
610       \glscrshortcuttrue
611       \DefineAcronymSynonyms
612     }%
613   \or % abbreviations
614     \renewcommand*{\@glstr@setupshortcuts}{%
615       \GlsXtrDefineAbbreviationShortcuts
616     }%
617   \or % abbr
618     \renewcommand*{\@glstr@setupshortcuts}{%
619       \GlsXtrDefineAbbreviationShortcuts
620     }%
621   \or % other

```

```

622 \renewcommand*{\@glxtr@setupshortcuts}{%
623 \GlsXtrDefineOtherShortcuts
624 }%
625 \or % all
626 \renewcommand*{\@glxtr@setupshortcuts}{%
627 \glsacrshortcutstrue
628 \GlsXtrDefineAcShortcuts
629 \GlsXtrDefineAbbreviationShortcuts
630 \GlsXtrDefineOtherShortcuts
631 }%
632 \or % true
633 \renewcommand*{\@glxtr@setupshortcuts}{%
634 \glsacrshortcutstrue
635 \GlsXtrDefineAcShortcuts
636 \GlsXtrDefineAbbreviationShortcuts
637 \GlsXtrDefineOtherShortcuts
638 }%
639 \or % ac
640 \renewcommand*{\@glxtr@setupshortcuts}{%
641 \glsacrshortcutstrue
642 \GlsXtrDefineAcShortcuts
643 }%

```

Leave none and false as last option.

```

644 \else % none, false
645 \renewcommand*{\@glxtr@setupshortcuts}{}%
646 \fi
647 }

```

`lsxtr@doaccsupp`

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

glossaries-accsupp can't be loaded after glossaries-extra. glossaries-accsupp v4.29+ checks `\@glxtr@doaccsupp` to determine if it's been loaded too late.

`accsupp` If `accsupp`, load `glossaries-accsupp` package.

```

649 \@glxtr@declareoption{accsupp}{%
650 \renewcommand*{\@glxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}

```

`tr@doloadprefix`

```
651 \newcommand*{\@glxtr@doloadprefix}{}
```

`prefix` If `prefix`, load `glossaries-prefix` package.

```

652 \@glxtr@declareoption{prefix}{%
653 \renewcommand*{\@glxtr@doloadprefix}{\RequirePackage{glossaries-prefix}}}

```


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

```
654 \newcommand{\glxtrNoGlossaryWarning}[1]{%
655   \GlossariesExtraWarning{Glossary '#1' is missing}%
656   \@glxtr@defaultnoglossarywarning{#1}%
657 }
```

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

```
658 \define@choicekey{glossaries-extra.sty}{nomissingglstext}
659 [ \@glxtr@nomissingglstextval \@glxtr@nomissingglstextnr ]%
660 {true,false}[true]{%
661   \ifcase \@glxtr@nomissingglstextnr \relax % true
662     \renewcommand{\glxtrNoGlossaryWarning}[1]{\null}%
663   \else % false
664     \renewcommand{\glxtrNoGlossaryWarning}[1]{%
665       \@glxtr@defaultnoglossarywarning{#1}%
666     }%
667   \fi
668 }
```

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

xtr@redefstyles

```
669 \newcommand*{\@glxtr@redefstyles}{}
```

stylemods

```
670 \define@key{glossaries-extra.sty}{stylemods}[default]{%
671   \ifstrequal{#1}{default}%
672   {%
673     \renewcommand*{\@glxtr@redefstyles}{%
674       \RequirePackage{glossaries-extra-stylemods}}%
675   }%
676   {%
677     \ifstrequal{#1}{all}%
678     {%
679       \renewcommand*{\@glxtr@redefstyles}{%
680         \PassOptionsToPackage{all}{glossaries-extra-stylemods}%
681         \RequirePackage{glossaries-extra-stylemods}%
682       }%
683     }%
684     {%
685       \renewcommand*{\@glxtr@redefstyles}{}%
686       \@for \@glxtr@tmp:=#1\do{%
687         \IfFileExists{glossary-\@glxtr@tmp.sty}%
688         {%
689           \eappto \@glxtr@redefstyles{%
690             \noexpand\RequirePackage{glossary-\@glxtr@tmp}}%
691         }%
692       }%
693     }%
694   }%
695 }
```

```

693         \PackageError{glossaries-extra}%
694         {Glossaries style package ‘glossary-@glxtr@tmp.sty’
695         doesn’t exist (did you mean to use the ‘style’ key?)}%
696         {The list of values (#1) in the ‘stylemods’ key should
697         match the glossary-xxx.sty files provided with
698         glossaries.sty}%
699     }%
700 }%
701 \appto\@glxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
702 }
703 }%
704 }

```

glxtr@do@style

```

705 \newcommand*{\@glxtr@do@style}{}

```

style Since the stylemods option can automatically load extra style packages, deal with the style option after those packages have been loaded.

```

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

```

Defer actual style change:

```

707 \renewcommand*{\@glxtr@do@style}{%

```

Set this as the default style:

```

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

```

Set this style:

```

709 \setglossarystyle{#1}%

```

```

710 }%

```

```

711 }

```

c@wrglossaryctr

Increments the associated counter if enabled. Does nothing by default. The optional argument is the entry label in case it’s required, but the wrglossary counter is globally used by all entries.

```

712 \newcommand*{\glxtr@inc@wrglossaryctr}[1]{}

```

cationHyperlink

```

\glxtrinternallocationhyperlink{<counter>}{<prefix>}{<location>}

```

The first two arguments are always control sequences.

```

713 \newcommand*{\GlsXtrInternalLocationHyperlink}[3]{%

```

```

714 \glxtrhyperlink{#1#2#3}{#3}%

```

```

715 }

```

cationhyperlink

```

716 \newcommand*{\@glxtr@wrglossary@locationhyperlink}[3]{%

```

```

717 \pageref{wrglossary.#3}%

```

```

718 }

```

`indexcounter` Define the `wrglossary` counter that's incremented every time an entry is indexed, except for cross-references. This is designed for use with `bib2gls v1.4+`. It can work with the other indexing methods but it will interfere with the number list collation. This option automatically implements `counter=wrglossary`.

Since `glossaries` automatically loads `amsmath`, there may be a problem if the indexing occurs in the equation environment, because only one `\label` is allowed in each instance of that environment. It's best to change the counter when in `maths` mode.

```

719 \@glxstr@declareoption{indexcounter}{%
720   \glxstr@doption{counter=wrglossary}%
721   \ifundef\c@wrglossary
722   {%
723     \newcounter{wrglossary}%
724     \renewcommand{\thewrglossary}{\arabic{wrglossary}}%
725   }%
726   {}%
727   \renewcommand*{\glxstr@inc@wrglossaryctr}[1]{%

```

Only increment if the current counter is `wrglossary`.

```

728   \ifdefstring\@glsc@counter{wrglossary}%
729   {%
730     \refstepcounter{wrglossary}%
731     \label{wrglossary.\thewrglossary}%
732   }%
733   {}%
734 }%
735 \renewcommand*{\GlsXtrInternalLocationHyperlink}[3]{%
736   \ifdefstring\glsc@entrycounter{wrglossary}%
737   {%
738     \@glxstr@wrglossary@locationhyperlink{##1}{##2}{##3}%
739   }%
740   {\glxstr@hyperlink{##1##2##3}{##3}}%
741 }%
742 }

```

`sxtrwrglossmark` Marks the place where indexing occurs. Does nothing by default.

```

743 \newcommand*{\@glxstrwrglossmark}{}

```

`sxtrwrglossmark` Since `\glscadd` can be used in the preamble, this action needs to be disabled until the start of the document.

```

744 \newcommand*{\@glxstrwrglossmark}{}
745 \AtBeginDocument{\renewcommand*{\@glxstrwrglossmark}{\@glxstrwrglossmark}}

```

`sxtrwrglossmark` Does nothing by default.

```

746 \newcommand*{\glxstrwrglossmark}{\ensuremath{\cdot}}

```

`tr@doshowtarget`

```

747 \newcommand\@glxstr@doshowtarget[2]{#2}

```

debug Provide extra debug options.

```

748 \define@choicekey{glossaries-extra.sty}{debug}
749 [\@glsxtr@debugval\@glsxtr@debugnr]%
750 {true,false,showtargets,showwrgloss,all,showaccsupp}[true]{%
751   \ifcase\@glsxtr@debugnr\relax % true
752   \glsxtr@doption{debug=true}%
753   \renewcommand*{\@glsxtrwrglossmark}{}%
754   \or % false
755   \glsxtr@doption{debug=false}%
756   \renewcommand*{\@glsxtrwrglossmark}{}%
757   \let\@glsxtr@doshowtarget\@secondoftwo
758   \or % showtargets
759   \glsxtr@doption{debug=showtargets}%
760   \def\@glsxtr@doshowtarget{\@glsxtr@showtargetleft}%
761   \or % showwrgloss
762   \glsxtr@doption{debug=true}%
763   \renewcommand*{\@glsxtrwrglossmark}{\glsxtrwrglossmark}%
764   \or % all
765   \glsxtr@doption{debug=showtargets,debug=showaccsupp}%
766   \renewcommand*{\@glsxtrwrglossmark}{\glsxtrwrglossmark}%
767   \def\@glsxtr@doshowtarget{\@glsxtr@showtargetleft}%
768   \or % showaccsupp
769   \glsxtr@doption{debug=showaccsupp}%
770   \fi
771 }

```

showtargetouter

```

772 \newcommand*{\glsxtr@showtargetouter}{\glsshowtargetouter}

```

showtargetinner

```

773 \newcommand*{\glsxtr@showtargetinner}[1]{\glsshowtargetinner{#1}}

```

Debugging show targets.

rshowtargetleft

```

774 \newcommand{\@glsxtr@showtargetleft}[2]{\@glsshowtarget{#1}#2\@glsxtr@showtargetmark}%

```

showtargetright

```

775 \newcommand{\@glsxtr@showtargetright}[2]{\@glsxtr@showtargetmark#2\@glsshowtarget{#1}}%

```

rshowtargetmark

```

776 \newcommand{\@glsxtr@showtargetmark}{}%

```

showtargets Implements debug=showtargets and provides extra adjustments.

```

777 \define@choicekey{glossaries-extra.sty}{showtargets}
778 [\@glsxtr@showtargetsval\@glsxtr@showtargetsnr]%
779 {left,right,innerleft,innerright,annoteleft,annoteright}%
780 {%

```

```

781 \glxtr@doption{debug=showtargets}%
782 \ifcase\@glxtr@showtargets\relax
783   \def\@glxtr@doshowtarget{\@glxtr@showtargetleft}%
784   \def\glxtr@showtargetouter{\glsshowtargetouter}%
785   \def\glxtr@showtargetinner{\glsshowtargetinner}%
786   \let\@glxtr@showtargetmark\empty
787 \or
788   \def\@glxtr@doshowtarget{\@glxtr@showtargetright}%
789   \def\glxtr@showtargetouter{\glsshowtargetouter}%
790   \def\glxtr@showtargetinner{\glsshowtargetinner}%
791   \let\@glxtr@showtargetmark\empty
792 \or
793   \def\@glxtr@doshowtarget{\@glxtr@showtargetleft}%
794   \def\glxtr@showtargetouter{\glxtr@showtargetinner}%
795   \def\glxtr@showtargetinner{\glsshowtargetinnersymleft}%
796   \let\@glxtr@showtargetmark\empty
797 \or
798   \def\@glxtr@doshowtarget{\@glxtr@showtargetright}%
799   \def\glxtr@showtargetouter{\glxtr@showtargetinner}%
800   \def\glxtr@showtargetinner{\glsshowtargetinnersymright}%
801   \let\@glxtr@showtargetmark\empty
802 \or
803   \def\@glxtr@doshowtarget{\@glxtr@showtargetleft}%
804   \def\glxtr@showtargetouter{\glxtr@showtargetinner}%
805   \def\glxtr@showtargetinner{\glsshowtargetinnersymleft}%
806   \def\@glxtr@showtargetmark{\@glsshowtargetmarkfmt\glxtr@showtargetsymbolright}%
807 \or
808   \def\@glxtr@doshowtarget{\@glxtr@showtargetright}%
809   \def\glxtr@showtargetouter{\glxtr@showtargetinner}%
810   \def\glxtr@showtargetinner{\glsshowtargetinnersymright}%
811   \def\@glxtr@showtargetmark{\@glsshowtargetmarkfmt\glxtr@showtargetsymbolleft}%
812 \fi
813 }

```

Pass all other options to glossaries.

```

814 \DeclareOptionX*{%
815   \expandafter\glxtr@doption\expandafter{\CurrentOption}}

```

Process options.

```
816 \ProcessOptionsX
```

Load glossaries if not already loaded.

```
817 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
818 \@glxtr@doaccsupp
```

Load the glossaries-prefix package if required.

```
819 \@glxtr@doloadprefix
```

Redefine \glspostdescription if required.

```
820 \@glxtr@defpostpunc
```

glsdoshowtarget Added to glossaries v4.50 so many not be defined. Need to redefine it so use \def.
821 \def\glsdoshowtarget{\@glstr@doshowtarget}

rgetssymbolright
822 \newcommand{\glstrshowtargetssymbolright}{\tiny\$\triangleleft\$}%

argetssymbolleft
823 \newcommand{\glstrshowtargetssymbolleft}{\tiny\$\triangleright\$}%

showtargetinner Only added to glossaries in v4.50 so may not be defined.
824 \providecommand*\glsshowtargetinner[1]{\glsshowtargetfont [#1]}

sshowtargetfont Only added to glossaries in v4.45 so may not be defined.
825 \providecommand*\glsshowtargetfont{\ttfamily\footnotesize}

rcontentsymleft
826 \newcommand*\glsshowtargetinnersymleft[1]{%
827 \glsshowtargetinner{#1}\allowbreak\glstrshowtargetssymbolleft}

contentsymright
828 \newcommand*\glsshowtargetinnersymright[1]{%
829 \glstrshowtargetssymbolright\allowbreak\glsshowtargetinner{#1}}

showtargetouter Only added to glossaries in v4.45 so may not be defined.
830 \providecommand*\glsshowtargetouter[1]{%
831 \glsshowtargetssymbol\marginpar{\glsshowtargetssymbol\glsshowtargetfont #1}}

\@glsshowtarget Only added to glossaries in v4.32 so may not be defined.
832 \providecommand*\@glsshowtarget[1]{}

\glsshowtarget This command was introduced to glossaries v4.32 so it may not be defined. Therefore it's defined here using \def. \glsshowtargetouter was introduced in glossaries v4.45, so that also may not be defined.
833 \def\glsshowtarget#1{%
834 \glstrtitleorpdforheading
835 {%
836 \ifmmode
837 \nfss@text{\glstrshowtargetinner{#1}}%
838 \else
839 \ifinner
840 \glstrshowtargetinner{#1}%
841 \else
842 \glstrshowtargetouter{#1}%
843 \fi
844 \fi
845 }%
846 {[#1]}%
847 {\protect\glsshowtargetinner{#1}}}%
848 }

owtargetmarkfmt

```
849 \newcommand*{\@glsshowtargetmarkfmt}[1]{%
850   \glstrtitleorpdforheading
851   {%
852     \ifmmode \nfss@text{#1}\else #1\fi
853   }%
854   {%
855     {\ifmmode \nfss@text{#1}\else #1\fi}%
856 }
```

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

```
857 \let\@glstr@org@doseeglossary\@do@seeglossary
```

r@doseeglossary This doesn't increment the associated counter.

```
858 \newcommand*{\@glstr@doseeglossary}[2]{%
859   \glsdofexists{#1}%
860   {%
861     \@glstrwrglossmark
862     \@glstr@org@doseeglossary{#1}{#2}%
863   }%
864 }
```

oindex@glossary

```
865 \newcommand*{\@glstr@dosee@alsoindex@glossary}[2]{%
866   \@glstr@recordsee{#1}{#2}%
867   \@glstr@doseeglossary{#1}{#2}%
868 }
```

@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.)

```
869 \let\@glstr@org@gloautosee\@glo@autosee
```

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

```
870 \if@glstr@autoseeindex
871 \else
872   \ifdef\@glstr@org@gloautosee
873   {%
874     {\PackageError{glossaries-extra}{‘autoseeindex=false’ package
875       option requires at least v4.30 of glossaries.sty}%
876     {You need to update the glossaries.sty package}%
877   }
878 \fi
```

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

```
879 \ifdef\@glo@autosee
880 {%
```

```

881 \renewcommand*{\@glo@autosee}{%
882 \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
883 }%
884 {}

```

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.

```

885 \renewcommand*{\glsc@checkseeallowed}{%
886 \if@glxtr@autoseeindex\@glsc@see@noindex\fi
887 }

```

Define abbreviations glossaries if required.

```

888 \@glxtr@abbreviationsdef
889 \let\@glxtr@abbreviationsdef\relax

```

Setup shortcuts if required.

```

890 \@glxtr@setupshortcuts

```

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

```

891 \@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:

```

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

```

Disable options that can only be used when the package is loaded:

```

893 \disable@keys{glossaries-extra.sty}{accsupp}

```

Now define the user command:

```

894 \newcommand*{\glossariesextrasetup}[1]{%
895 \let\glxtr@setup@record\relax
896 \let\@glxtr@setupshortcuts\relax
897 \let\@glxtr@redef@for@gl@sentries\relax
898 \let\@glxtr@doloadprefix\relax
899 \setkeys{glossaries-extra.sty}{#1}%
900 \@glxtr@abbreviationsdef
901 \let\@glxtr@abbreviationsdef\relax
902 \@glxtr@setupshortcuts
903 \glxtr@setup@record
904 \@glxtr@redef@for@gl@sentries
905 \@glxtr@doloadprefix
906 }

```

@@do@wrglossary Save original definition of @@do@wrglossary.

```

907 \let\glxtr@org@@do@wrglossary\@@do@wrglossary

```

@@do@wrglossary The new version adds code that can show a marker for debugging and increments the associated counter if enabled.

```

908 \newcommand*{\glxtr@@do@wrglossary}[1]{%
909 \@@glxtrwrglossmark

```



```

910 \glxstr@inc@wrglossaryctr{#1}%
911 \glxstr@org@@do@wrglossary{#1}%
912 }

```

`\saveentrycounter` Save original definition of `\@gls@saveentrycounter`.

```

913 \let\glxstr@saveentrycounter\@gls@saveentrycounter

```

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

```

914 \let\@gls@saveentrycounter\glxstr@indexonly@saveentrycounter

```

`\etcounterprefix` This command is provided by the base glossaries package, but is redefined here. The standard indexing methods don't directly store the hypertarget but instead need to split it into the counter, prefix and location parts, which can be reconstituted in the location list. Unfortunately, not all targets are in this form, so the links fail. With `record=nameref`, the complete target name can be saved, so this modification adjusts the warning.

```

915 \renewcommand*\@gls@getcounterprefix[2]{%
916   \protected@edef\@gls@thisloc{#1}\protected@edef\@gls@thisHloc{#2}%
917   \ifx\@gls@thisloc\@gls@thisHloc
918     \def\@glo@counterprefix{}%
919   \else
920     \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
921       \def\@glo@tmp{##2}%
922       \ifx\@glo@tmp\@empty
923         \def\@glo@counterprefix{}%
924       \else
925         \def\@glo@counterprefix{##1}%
926       \fi
927     }%
928     \@gls@get@counterprefix#2.#1\end@getprefix

```

Warn if no prefix can be formed, unless `record=nameref`.

```

929   \ifx\@glo@counterprefix\@empty
930     \ifx\glxstr@record@setting\glxstr@record@setting@nameref
931     \else
932       \GlossariesExtraWarning{Hyper target '#2' can't be formed by
933         prefixing^^Jlocation '#1'. You need to modify the
934         definition of \string\theH\@gls@counter^^Jotherwise you
935         will get the warning: "name{\@gls@counter.#1}' has been^^J
936         referenced but does not exist"%
937       \ifx\glxstr@record@setting\glxstr@record@setting@only
938         . You may want to consider using record=nameref instead%
939       \fi}%
940   \fi
941 \fi
942 \fi
943 }

```

Provide script dialect hook (does nothing unless redefined by `glossaries-extra-bib2gls`).

sxtrdialecthook

```
944 \newcommand*{\@glxsxtrdialecthook}{}
```

Set up record option if required.

```
945 \glxsxtr@setup@record
```

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

```
946 \AtBeginDocument{%
947   \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
948   \def\@glxsxtrundeftag{\glxsxtrundeftag}%
949 }
```

1.2 Extra Utilities

usedOrUndefined

```
\GlsXtrIfUnusedOrUndefined{<label>}{<true>}{<false>}
```

Does *<true>* if the entry given by *<label>* is either undefined or hasn't been used (or has had the first use flag reset).

```
950 \newcommand*{\GlsXtrIfUnusedOrUndefined}[3]{%
951   \ifglseentryexists{#1}%
952   {\ifbool{glo@\glsetoklabel{#1}@flag}{#3}{#2}}%
953   {#2}%
954 }
```

Starred form of `\ifglossaryexists` was only introduced to glossaries v4.46 so provide it if it hasn't been defined.

```
955 \ifdef\s@ifglossaryexists
956 {}
957 {
```

fglossaryexists

```
958   \renewcommand{\ifglossaryexists}{%
959     \@ifstar\s@ifglossaryexists\@ifglossaryexists
960   }
```

fglossaryexists

```
961   \newcommand{\@ifglossaryexists}[3]{%
962     \ifcsundef{@glotype@#1@out}{#3}{#2}%
963   }
```

fglossaryexists

```
964   \newcommand{\s@ifglossaryexists}[3]{%
965     \ifcsundef{glolist@#1}{#3}{#2}%
966   }
```

967 }

ifemptyglossary

`\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.)

```
968 \newcommand{\glxtrifemptyglossary}[3]{%
969   \ifcsdef{glolist@#1}%
970   {%
971     \ifcsstring{glolist@#1}{,}{#2}{#3}%
972   }%
973   {%
974     \glxtrundefaction{Glossary type '#1' doesn't exist}{}%
975     #2%
976   }%
977 }
```

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

```
978 \newcommand*\glxtrifkeydefined[3]{%
979   \key@ifundefined{glossentry}{#1}{#3}{#2}%
980 }
```

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

```
981 \newcommand*\glxtrprovidestoragekey{%
982   \@ifstar\@sglxtr@provide@storagekey\@glxtr@provide@storagekey
983 }
```

vide@storagekey Unstarred version.

```
984 \newcommand*\@glxtr@provide@storagekey[3]{%
985   \key@ifundefined{glossentry}{#1}%
986   {%
987     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
988     \appto\@glx@keymap{,}{#1}{#1}}%
989     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
990     \appto\@newglossaryentryposthook{%
991       \letcs{\@glo@tmp}{@glo@#1}%
992       \glx@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
993     }%
```

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

```
994   \ifblank{#3}
```

```

995     {}%
996     {%
997         \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
998     }%
999 }%
1000 {%

    Provide the no-link command if not already defined.

1001     \ifblank{#3}
1002     {}%
1003     {%
1004         \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1005     }%
1006 }%
1007 }

```

vide@storagekey Starred version.

```

1008 \newcommand*{\s@glxtr@provide@storagekey}[1]{%
1009     \key@ifundefined{glossentry}{#1}%
1010     {%
1011         \expandafter\newcommand\expandafter*\expandafter
1012         {\csname gls@assign@#1@field\endcsname}[2]{%
1013             \@gls@expand@field{##1}{#1}{##2}%
1014         }%
1015     }%
1016     {}%
1017     \@glxtr@provide@addstoragekey{#1}%
1018 }

```

The name of a text-block control sequence can be stored in a field (given by `\GlsXtrFmtField`). This command can then be used with `\glxtrfmt [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`

```

1019 \newcommand{\GlsXtrFmtField}{useri}

```

`tDefaultOptions`

```

1020 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

`\glxtrfmt` The post-link hook isn't done. This now has a starred form that checks for a final optional argument.

```

1021 \newrobustcmd*{\glxtrfmt}{\@ifstar\s@glxtrfmt\@glxtrfmt}

```

`\@glxtrfmt` Unstarred form.

```

1022 \newcommand*{\@glxtrfmt}[3][[]]{\@@glxtrfmt{#1}{#2}{#3}{}}

```

`\s@glstrfmt` Starred form.

```
1023 \newcommand*{\s@glstrfmt}[3][]{%
1024   \new@ifnextchar[{\s@glstrfmt{#1}{#2}{#3}}%
1025   {\s@glstrfmt{#1}{#2}{#3}{}}%
1026 }
```

`\s@@glstrfmt` Pick up final optional argument.

```
1027 \def\s@@glstrfmt#1#2#3[#4]{\s@glstrfmt{#1}{#2}{#3}{#4}}
```

`\@@glstrfmt` Actual inner working.

```
1028 \newcommand*{\@@glstrfmt}[4]{%
```

Since there's no post-link hook to worry about, grouping can be added to provide some protection against nesting (but in general nested link text should be avoided).

```
1029   \begingroup
1030     \def\glslabel{#2}%
1031     \glsd@ifexists{#2}%
1032     {%
1033       \ifglshasfield{\GlsXtrFmtField}{#2}%
1034       {%
1035         \let\do@glsl@link@checkfirsthyper\relax
1036         \expandafter\@glsl@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
1037         {\glstrfmtdisplay{\glscurrentfieldvalue}{#3}{#4}}%
1038       }%
1039       {\glstrfmtdisplay{@firstofone}{#3}{#4}}%
1040     }%
1041     {%
```

Has the default `noindex` been counteracted? If so, this needs `\glssadd` in case `bib2gls` needs to pick up the record.

```
1042     \begingroup
1043       \@glsl@setdefault@glslink@opts
1044       \setkeys{glslink}{\GlsXtrFmtDefaultOptions,#1}%
1045       \ifKV@glslink@noindex\else\glssadd{#2}\fi
1046     \endgroup
1047     \glstrfmtdisplay{@firstofone}{#3}{#4}%
1048   }%
1049 \endgroup
1050 }
```

`\glstrfmtdisplay` The command used internally by `\glstrfmt` to do the actual formatting. The first argument is the control sequence name, the second is the control sequence's argument, the third is the inserted material (if starred form used).

```
1051 \newcommand{\glstrfmtdisplay}[3]{\csuse{#1}{#2}#3}
```

`\glstrentryfmt` No link or indexing.

```
1052 \ifdef\texorpdfstring
1053 {
1054   \newcommand*{\glstrentryfmt}[2]{%
```

```

1055 \texorpdfstring{\@glxxtrententryfmt{#1}{#2}}{\@glxxttrpdfentryfmt{#1}{#2}}%
1056 }
1057 }
1058 {
1059 \newcommand*{\glxxtrententryfmt}{\@glxxtrententryfmt}
1060 }

```

`xttrpdfentryfmt` Use for the PDF bookmarks.

```

1061 \newcommand*{\glxxttrpdfentryfmt}[2]{#2}

```

`@glxxtrententryfmt`

```

1062 \newrobustcmd*{\@glxxtrententryfmt}[2]{%

```

Locally define `\glslabel` in case the helper command needs to access the label.

```

1063 {%
1064 \protected@edef\glslabel{#1}%
1065 \glsdoidexistsordo{#1}%
1066 {%
1067 \ifglshasfield{\GlsXtrFmtField}{#1}%
1068 {%
1069 \csuse{\glscurrentfieldvalue}{#2}%
1070 }%
1071 {#2}%
1072 }%
1073 {#2}%
1074 }%
1075 }

```

`xttrfieldlistadd` 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.

```

1076 \newcommand*{\glxxttrfieldlistadd}[3]{%
1077 \listcsadd{glo@\glsdetoklabel{#1}@#2}{#3}%
1078 }

```

`trfieldlistgadd` Similarly but uses `\listcsgadd`.

```

1079 \newcommand*{\glxxttrfieldlistgadd}[3]{%
1080 \listcsgadd{glo@\glsdetoklabel{#1}@#2}{#3}%
1081 }

```

`trfieldlistead` Similarly but uses `\listcseadd`.

```

1082 \newcommand*{\glxxttrfieldlistead}[3]{%
1083 \listcseadd{glo@\glsdetoklabel{#1}@#2}{#3}%
1084 }

```

`trfieldlistxadd` Similarly but uses `\listcsxadd`.

```

1085 \newcommand*{\glxxttrfieldlistxadd}[3]{%
1086 \listcsxadd{glo@\glsdetoklabel{#1}@#2}{#3}%
1087 }

```

Now provide commands to iterate over these lists.

fielddolistloop

```
1088 \newcommand*{\glxtrfielddolistloop}[2]{%
1089   \dolistcsloop{glo@\glsdetoklabel{#1}@#2}%
1090 }
```

fieldforlistloop

```
1091 \newcommand*{\glxtrfieldforlistloop}[3]{%
1092   \forlistcsloop{#3}{glo@\glsdetoklabel{#1}@#2}%
1093 }
```

fieldformatlist

```
1094 \newrobustcmd*{\glxtrfieldformatlist}[2]{%
1095   \begingroup
1096   \def\@dtl@formatlist@itemsep{}%
1097   \def\@dtl@formatlist@lastitem{}%
1098   \def\@dtl@formatlist@prelastitem{}%
1099   \def\@dtl@formatlist@prelastitemsep{}%
1100   \forlistcsloop{\@dtl@formatlist@handler}{glo@\glsdetoklabel{#1}@#2}%
1101   \@dtl@formatlist@prelastitem\@dtl@formatlist@lastitem
1102   \endgroup
1103 }
```

List element tests:

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

```
1104 \newcommand*{\glxtrfieldifinlist}[5]{%
1105   \ifinlistcs{#3}{glo@\glsdetoklabel{#1}@#2}{#4}{#5}%
1106 }
```

rfieldxifinlist Expands item.

```
1107 \newcommand*{\glxtrfieldxifinlist}[5]{%
1108   \xifinlistcs{#3}{glo@\glsdetoklabel{#1}@#2}{#4}{#5}%
1109 }
```

sxtrforcsvfield

`\glxtrforcsvfield{<label>}{<field>}{<cs handler>}`

```
1110 \newcommand*{\glxtrforcsvfield}{%
1111   \@ifstar\s@glxtrforcsvfield\@glxtrforcsvfield
1112 }
```

sxtrforcsvfield Unstarred version.

```
1113 \newcommand*{\@glxtrforcsvfield}[3]{%
```

```

1114 \@glsxtrifhasfield{#2}{#1}%
1115 {%
1116   \let\glsxtrendfor\@endfortrue
1117   \@for\@glsxtr@label:=\glscurrentfieldvalue\do
1118     {\expandafter#3\expandafter{\@glsxtr@label}}}%
1119 }%
1120 }

```

`sxtrforcsvfield` Starred version.

```

1121 \newcommand*{\s@glsxtrforcsvfield}[3]{%
1122   \s@glsxtrifhasfield{#2}{#1}%
1123   {%
1124     \let\glsxtrendfor\@endfortrue
1125     \@for\@glsxtr@label:=\glscurrentfieldvalue\do
1126       {\expandafter#3\expandafter{\@glsxtr@label}}}%
1127   }%
1128 }

```

`ldformatcsvlist`

```

1129 \newrobustcmd*{\glsxtrfieldformatcsvlist}[2]{%
1130   \@glsxtrifhasfield{#2}{#1}%
1131   {\@dtlformatlist\glscurrentfieldvalue}%
1132   }%
1133 }

```

`dValueInCsvList`

$\backslash\text{GlsXtrIfFieldValueInCsvList}\langle label\rangle\langle field\rangle\langle list\rangle\langle true\rangle\langle false\rangle$
--

```

1134 \newcommand*{\GlsXtrIfFieldValueInCsvList}{%
1135   \ifstar\s@GlsXtrIfFieldValueInCsvList\@GlsXtrIfFieldValueInCsvList
1136 }

```

Note `\DTLifinlist` performs one level on the list but not the element.

`dValueInCsvList` Unstarred version.

```

1137 \newcommand*{\@GlsXtrIfFieldValueInCsvList}[5]{%
1138   \@glsxtrifhasfield{#2}{#1}%
1139   {%
1140     \expandafter\DTLifinlist\expandafter{\glscurrentfieldvalue}%
1141     {#3}{#4}{#5}%
1142   }%
1143   {#5}%
1144 }

```

`dValueInCsvList` Starred version.

```

1145 \newcommand*{\s@GlsXtrIfFieldValueInCsvList}[5]{%

```



```

1146 \s@glstrifhasfield{#2}{#1}%
1147 {%
1148   \expandafter\DTLifinlist\expandafter{\glscurrentfieldvalue}%
1149   {#3}{#4}{#5}%
1150 }%
1151 {#5}%
1152 }

```

eInFieldCsvList

```
\GlsXtrIfValueInFieldCsvList{<label>}{<field>}{<value>}{<true>}{<false>}
```

Essentially the reverse. Tests if the given value is in the given field which should contain a comma-separated list.

```

1153 \newcommand*{\GlsXtrIfValueInFieldCsvList}{%
1154   \@ifstar\s@GlsXtrIfValueInFieldCsvList\@GlsXtrIfValueInFieldCsvList
1155 }

```

eInFieldCsvList Unstarred version.

```

1156 \newcommand*{\@GlsXtrIfValueInFieldCsvList}[5]{%
1157   \@glstrifhasfield{#2}{#1}%
1158   {%
1159     \DTLifinlist{#3}{\glscurrentfieldvalue}{#4}{#5}%
1160   }%
1161   {#5}%
1162 }

```

eInFieldCsvList Unstarred version.

```

1163 \newcommand*{\s@GlsXtrIfValueInFieldCsvList}[5]{%
1164   \s@glstrifhasfield{#2}{#1}%
1165   {%
1166     \DTLifinlist{#3}{\glscurrentfieldvalue}{#4}{#5}%
1167   }%
1168   {#5}%
1169 }

```

eInFieldCsvList

```
\xGlsXtrIfValueInFieldCsvList{<label>}{<field>}{<value>}{<true>}{<false>}
```

As above but fully expand *<value>*.

```

1170 \newcommand*{\xGlsXtrIfValueInFieldCsvList}{%
1171   \@ifstar\s@xGlsXtrIfValueInFieldCsvList\xGlsXtrIfValueInFieldCsvList
1172 }

```

eInFieldCsvList Unstarred version.

```

1173 \newcommand*{\@xGlsXtrIfValueInFieldCsvList}[5]{%
1174   \@glstrifhasfield{#2}{#1}%
1175   {%
1176     \protected@edef\@gls@tmp{#3}%
1177     \expandafter\DTLifinlist\expandafter{\@gls@tmp}{\glscurrentfieldvalue}{#4}{#5}%
1178   }%
1179   {#5}%
1180 }

```

eInFieldCsvList Unstarred version.

```

1181 \newcommand*{\s@xGlsXtrIfValueInFieldCsvList}[5]{%
1182   \s@glstrifhasfield{#2}{#1}%
1183   {%
1184     \protected@edef\@gls@tmp{#3}%
1185     \expandafter\DTLifinlist\expandafter{\@gls@tmp}{\glscurrentfieldvalue}{#4}{#5}%
1186   }%
1187   {#5}%
1188 }

```

lsxtrifhasfield

`\glstrifhasfield{<field>}{<label>}{<true>}{<false>}`

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.

```

1189 \newrobustcmd{\glstrifhasfield}{%
1190   \@ifstar{\s@glstrifhasfield}{\glstrifhasfield}%
1191 }

```

lsxtrifhasfield Unstarred version adds grouping.

```

1192 \newcommand{\@glstrifhasfield}[4]{%
1193   {\s@glstrifhasfield{#1}{#2}{#3}{#4}}%
1194 }

```

lsxtrifhasfield Starred version omits grouping.

```

1195 \newcommand{\s@glstrifhasfield}[4]{%
1196   \letcs{\glscurrentfieldvalue}{glo@\glsetoklabel{#2}@#1}%
1197   \ifundef\glscurrentfieldvalue
1198     {#4}%
1199   {%
1200     \ifdefempty\glscurrentfieldvalue{#4}{#3}%
1201   }%
1202 }

```

rIfFieldNonZero Designed for numeric fields.

```
1203 \newcommand{\GlsXtrIfFieldNonZero}{%  
1204   \@ifstar\s@GlsXtrIfFieldNonZero\@GlsXtrIfFieldNonZero  
1205 }
```

rIfFieldNonZero

```
1206 \newcommand{\@GlsXtrIfFieldNonZero}[4]{%  
1207   \@GlsXtrIfFieldCmpNum{#1}{#2}{=}{0}{#4}{#3}%  
1208 }
```

rIfFieldNonZero

```
1209 \newcommand{\s@GlsXtrIfFieldNonZero}[4]{%  
1210   \s@GlsXtrIfFieldCmpNum{#1}{#2}{=}{0}{#4}{#3}%  
1211 }
```

XtrIfFieldEqNum

$\backslash\text{GlsXtrIfFieldEqNum}\{\langle field \rangle\}\{\langle label \rangle\}\{\langle value \rangle\}\{\langle true \rangle\}\{\langle false \rangle\}$

Designed for numeric fields.

```
1212 \newcommand{\GlsXtrIfFieldEqNum}{%  
1213   \@ifstar\s@GlsXtrIfFieldEqNum\@GlsXtrIfFieldEqNum  
1214 }
```

XtrIfFieldEqNum

```
1215 \newcommand{\@GlsXtrIfFieldEqNum}[5]{%  
1216   \@GlsXtrIfFieldCmpNum{#1}{#2}{=}{#3}{#4}{#5}%  
1217 }
```

XtrIfFieldEqNum

```
1218 \newcommand{\s@GlsXtrIfFieldEqNum}[5]{%  
1219   \s@GlsXtrIfFieldCmpNum{#1}{#2}{=}{#3}{#4}{#5}%  
1220 }
```

trIfFieldCmpNum

$\backslash\text{GlsXtrIfFieldCmpNum}\{\langle field \rangle\}\{\langle label \rangle\}\{\langle comparison \rangle\}\{\langle value \rangle\}\{\langle true \rangle\}\{\langle false \rangle\}$

Designed for numeric fields.

```
1221 \newcommand{\GlsXtrIfFieldCmpNum}{%  
1222   \@ifstar\s@GlsXtrIfFieldCmpNum\@GlsXtrIfFieldCmpNum  
1223 }
```

trIfFieldCmpNum

```

1224 \newcommand{\@GlsXtrIfFieldCmpNum}[6]{%
1225   {%
1226     \letcs{\glscurrentfieldvalue}{glo@\glsdetoklabel{#2}@#1}%
1227     \ifundef\glscurrentfieldvalue
1228     {\def\glscurrentfieldvalue{0}}%
1229     {%
1230       \ifdefempty\glscurrentfieldvalue
1231       {\def\glscurrentfieldvalue{0}}%
1232       {%
1233       }%
1234       \ifnum\glscurrentfieldvalue#3#4\relax #5\else #6\fi
1235     }%
1236 }

```

trIfFieldCmpNum

```

1237 \newcommand{\s@GlsXtrIfFieldCmpNum}[6]{%
1238   \letcs{\glscurrentfieldvalue}{glo@\glsdetoklabel{#2}@#1}%
1239   \ifundef\glscurrentfieldvalue
1240   {\def\glscurrentfieldvalue{0}}%
1241   {%
1242     \ifdefempty\glscurrentfieldvalue
1243     {\def\glscurrentfieldvalue{0}}%
1244     {%
1245     }%
1246     \ifnum\glscurrentfieldvalue#3#4\relax #5\else #6\fi
1247 }

```

XtrIfFieldUndef

`\GlsXtrIfFieldUndef{<field>}{<label>}{<true>}{<false>}`

Just uses `\ifcsundef`.

```

1248 \newcommand{\GlsXtrIfFieldUndef}[2]{%
1249   \ifcsundef{glo@\glsdetoklabel{#2}@#1}%
1250 }

```

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

```

1251 \newcommand*\glsxtrusefield[2]{%
1252   \@gls@entry@field{#1}{#2}%
1253 }

```

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

```

1254 \ifdef\texorpdfstring
1255 {
1256   \newcommand*\Glsxtrusefield[2]{%

```

```

1257 \texorpdfstring
1258 {\@Gls@entry@field{#1}{#2}}
1259 {\@gls@entry@field{#1}{#2}}%
1260 }
1261 }
1262 {
1263 \newcommand*{\Glsxtrusefield}[2]{%
1264 \@Gls@entry@field{#1}{#2}%
1265 }
1266 }

```

`\Glsxtrusefield` As above but convert to all caps.

```

1267 \ifdef\texorpdfstring
1268 {
1269 \newcommand*{\Glsxtrusefield}[2]{%
1270 \texorpdfstring
1271 {\glsdoifexists{#1}{\mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}}}%
1272 {\@gls@entry@field{#1}{#2}}%
1273 }
1274 }
1275 {
1276 \newcommand*{\Glsxtrusefield}[2]{%
1277 \glsdoifexists{#1}{\mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}}%
1278 }
1279 }

```

`entryparentname`

```

1280 \newcommand*{\glsxtrentryparentname}[1]{%
1281 \ifcsdef{glo@glstdetoklabel{#1}@parent}%
1282 {\csuse{glo@csuse{glo@glstdetoklabel{#1}@parent}@name}}%
1283 {}%
1284 }

```

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

```

1285 \newcommand*{\glsxtrdeffield}[2]{\csdef{glo@glstdetoklabel{#1}@#2}}

```

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

```

1286 \newcommand*{\glsxtredeffield}[2]{\protected@csdef{glo@glstdetoklabel{#1}@#2}}

```

`trapptocsvfield` Similar to the above but will append value with a leading comma if the field is already defined. This is used by **bib2gls**. There's no check if the entry has been defined. (Because of the way that bib2gls's save-from-alias etc options are implemented, the entry may not have yet been written to the glstex file when this command is used.)

```

1287 \newcommand*{\glsxtrapptocsvfield}[3]{%
1288 \ifcsdef{glo@glstdetoklabel{#1}@#2}%
1289 {\csappto{glo@glstdetoklabel{#1}@#2}{, #3}}%
1290 {\csdef{glo@glstdetoklabel{#1}@#2}{#3}}%
1291 }

```

etfieldifexists

```
1292 \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.

```
1293 \newrobustcmd*{\GlsXtrSetField}[3]{%
1294   \glstrsetfieldifexists{#1}{#2}%
1295   {\csdef{glo@\glsetoklabel{#1}@#2}{#3}}%
1296 }
```

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

```
1297 \newrobustcmd*{\GlsXtrLetField}[3]{%
1298   \glstrsetfieldifexists{#1}{#2}%
1299   {\cslet{glo@\glsetoklabel{#1}@#2}{#3}}%
1300 }
```

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

```
1301 \newrobustcmd*{\csGlsXtrLetField}[3]{%
1302   \glstrsetfieldifexists{#1}{#2}%
1303   {\csletcs{glo@\glsetoklabel{#1}@#2}{#3}}%
1304 }
```

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.

```
1305 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
1306   \glstrsetfieldifexists{#1}{#2}%
1307   {\csletcs{glo@\glsetoklabel{#1}@#2}{glo@\glsetoklabel{#3}@#4}}%
1308 }
```

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

```
1309 \newrobustcmd*{\gGlsXtrSetField}[3]{%
1310   \glstrsetfieldifexists{#1}{#2}%
1311   {\csgdef{glo@\glsetoklabel{#1}@#2}{#3}}%
1312 }
```

xGlsXtrSetField

```
1313 \newrobustcmd*{\xGlsXtrSetField}[3]{%
1314   \glstrsetfieldifexists{#1}{#2}%
1315   {\protected@csxdef{glo@\glsetoklabel{#1}@#2}{#3}}%
1316 }
```

eGlsXtrSetField

```
1317 \newrobustcmd*{\eGlsXtrSetField}[3]{%
1318   \glstrsetfieldifexists{#1}{#2}%
1319   {\protected@csedef{glo@\glsetoklabel{#1}@#2}{#3}}%
1320 }
```

XtrIfFieldEqStr Starred version uses starred version of \glxtrifhasfield (that is, no grouping).

```
1321 \newcommand*{\GlsXtrIfFieldEqStr}{%
1322   \@ifstar\s@GlsXtrIfFieldEqStr\@GlsXtrIfFieldEqStr
1323 }
```

XtrIfFieldEqStr

```
1324 \newrobustcmd*{\@GlsXtrIfFieldEqStr}[5]{%
1325   \@glxtrifhasfield{#1}{#2}%
1326   {%
1327     \ifdefstring{\glscurrentfieldvalue}{#3}{#4}{#5}%
1328   }%
1329   {#5}%
1330 }
```

XtrIfFieldEqStr

```
1331 \newrobustcmd*{\s@GlsXtrIfFieldEqStr}[5]{%
1332   \s@glxtrifhasfield{#1}{#2}%
1333   {%
1334     \ifdefstring{\glscurrentfieldvalue}{#3}{#4}{#5}%
1335   }%
1336   {#5}%
1337 }
```

rIfFieldEqXpStr Like the above but first expands the string. Starred version uses starred version of \glxtrifhasfield (that is, no grouping).

```
1338 \newcommand*{\GlsXtrIfFieldEqXpStr}{%
1339   \@ifstar\s@GlsXtrIfFieldEqXpStr\@GlsXtrIfFieldEqXpStr
1340 }
```

rIfFieldEqXpStr

```
1341 \newrobustcmd*{\@GlsXtrIfFieldEqXpStr}[5]{%
1342   \@glxtrifhasfield{#1}{#2}%
1343   {%
1344     \protected@edef\@gls@tmp{#3}%
1345     \ifdefequal{\glscurrentfieldvalue}{\@gls@tmp}{#4}{#5}%
1346   }%
1347   {#5}%
1348 }
```

rIfFieldEqXpStr

```
1349 \newrobustcmd*{\s@GlsXtrIfFieldEqXpStr}[5]{%
1350   \s@glxtrifhasfield{#1}{#2}%
1351   {%
1352     \protected@edef\@gls@tmp{#3}%
1353     \ifdefequal{\glscurrentfieldvalue}{\@gls@tmp}{#4}{#5}%
1354   }%
1355   {#5}%
1356 }
```

fXpFieldEqXpStr Like the above but also expands the field value. Starred version uses starred version of `\glxtrifhasfield` (that is, no grouping).

```
1357 \newcommand*{\GlsXtrIfXpFieldEqXpStr}{%
1358   \@ifstar\s@GlsXtrIfXpFieldEqXpStr\@GlsXtrIfXpFieldEqXpStr
1359 }
```

fXpFieldEqXpStr

```
1360 \newrobustcmd*{\@GlsXtrIfXpFieldEqXpStr}[5]{%
1361   \glxtrifhasfield{#1}{#2}%
1362   {%
1363     \protected@edef\@gls@tmp{\glscurrentfieldvalue}%
1364     \let\glscurrentfieldvalue\@gls@tmp
1365     \protected@edef\@gls@tmp{#3}%
1366     \ifdefequal{\glscurrentfieldvalue}{\@gls@tmp}{#4}{#5}%
1367   }%
1368   {#5}%
1369 }
```

fXpFieldEqXpStr

```
1370 \newrobustcmd*{\s@GlsXtrIfXpFieldEqXpStr}[5]{%
1371   \s@glxtrifhasfield{#1}{#2}%
1372   {%
1373     \protected@edef\@gls@tmp{\glscurrentfieldvalue}%
1374     \let\glscurrentfieldvalue\@gls@tmp
1375     \protected@edef\@gls@tmp{#3}%
1376     \ifdefequal{\glscurrentfieldvalue}{\@gls@tmp}{#4}{#5}%
1377   }%
1378   {#5}%
1379 }
```

sXtrForeignText

`\GlsXtrForeignText{<entry label>}{<text>}`

If a field is used to store a language tag (such as en-GB or de-CH-1996) then this command uses `tracklang`'s interface to encapsulate *<text>*. The field identifying the locale is given by `\GlsXtrForeignTextField`.

```
1380 \ifdef\foreignlanguage
1381 {
1382   \ifdef\GetTrackedDialectFromLanguageTag
1383   {
1384     \newcommand{\GlsXtrForeignText}[2]{%
```

In case this is used inside the argument of `\glxtrifhasfield`, save and restore `\glscurrentfieldvalue`.

```
1385     \let\@glxtr@org@currentfieldvalue\glscurrentfieldvalue
1386     \glxtrifhasfield{\GlsXtrForeignTextField}{#1}%
1387     {%
```



```

1388 \expandafter\GetTrackedDialectFromLanguageTag\expandafter
1389 {\glscurrentfieldvalue}\@glsxtr@dialect}%
1390 \let\@glsxtr@locale\glscurrentfieldvalue
1391 \let\glscurrentfieldvalue\@glsxtr@org@currentfieldvalue
1392 \ifdefempty\@glsxtr@dialect
1393 {%

```

An exact match hasn't been found. A partial match can only be obtained with at least tracklang v1.3.6.

```

1394 \ifundef\TrackedDialectClosestSubMatch
1395 {%
1396 \GlossariesExtraWarning{Can't obtain dialect label
1397 (tracklang v1.3.6+ required)}}%
1398 }%
1399 {\let\@glsxtr@dialect\TrackedDialectClosestSubMatch}%
1400 }%
1401 {}%
1402 \ifdefempty\@glsxtr@dialect
1403 {%

```

No tracked dialect found for the root language.

```

1404 }%
1405 {%

```

Check if there's a caption hook for the given dialect label.

```

1406 \ifcsundef{captions\@glsxtr@dialect}{}%
1407 {%

```

Dialect label not recognised. Check if there's a known mapping.

```

1408 \IfTrackedDialectHasMapping{\@glsxtr@dialect}%
1409 {%
1410 \edef\@glsxtr@dialect{%
1411 \GetTrackedDialectToMapping{\@glsxtr@dialect}}%

```

Does a caption hook exist for this?

```

1412 \ifcsundef{captions\@glsxtr@dialect}{}%
1413 {%

```

No mapping. Try root language label instead.

```

1414 \ifcsundef{captions\@tracklang@lang}{}%
1415 {%
1416 \let\@glsxtr@dialect\@tracklang@lang
1417 }%
1418 }%
1419 }%
1420 {%

```

No mapping. Try root language label instead.

```

1421 \ifcsundef{captions\@tracklang@lang}{}%
1422 {%
1423 \let\@glsxtr@dialect\@tracklang@lang
1424 }%

```

```

1425         }%
1426     }%
1427 }%
1428 \ifdefempty\@glstr@dialect
1429 {%
1430     \GlsXtrUnknownDialectWarning{\@glstr@locale}{\@tracklang@lang}%
1431     #2%
1432 }%
1433 {\foreignlanguage{\@glstr@dialect}{#2}}%
1434 }%
1435 {#2}% key not set
1436 }
1437 }
1438 {
1439     \newcommand{\GlsXtrForeignText}[2]{%
1440         \GlossariesExtraWarning{Can't encapsulate foreign text:
1441             tracklang v1.3.6+ required}%
1442         #2%
1443     }
1444 }
1445 }
1446 {
    \foreignlanguage isn't defined so just do <text>.
1447 \newcommand{\GlsXtrForeignText}[2]{#2}
1448 }

```

`\foreignTextField` This is the `user2` field by default but may be redefined as required.

```

1449 \newcommand*{\GlsXtrForeignTextField}{userii}

```

`\nDialectWarning`

```

1450 \newcommand*{\GlsXtrUnknownDialectWarning}[2]{%
1451     \GlossariesExtraWarning{Can't determine valid dialect label
1452         for locale '#1' (root language: #2)}%
1453 }

```

`\glstrpageref` Like `\glssrefentry` but references the page number instead (if entry counting is on). The base glossaries package only introduced `\GlsEntryCounterLabelPrefix` in version 4.38, so it may not be defined.

```

1454 \ifdef\GlsEntryCounterLabelPrefix
1455 {%
1456     \newcommand*{\glstrpageref}[1]{%
1457         \ifglssentrycounter
1458             \pageref{\GlsEntryCounterLabelPrefix\glsdetoklabel{#1}}%
1459         \else
1460             \ifglssubentrycounter
1461                 \pageref{\GlsEntryCounterLabelPrefix\glsdetoklabel{#1}}%
1462             \else
1463                 \gls{#1}%

```

```

1464     \fi
1465     \fi
1466   }
1467 }%
1468 {%
1469   \newcommand*{\glstrpageref}[1]{%
1470     \ifglentrycounter
1471       \pageref{glentry-\glstetoklabel{#1}}%
1472     \else
1473       \ifglssubentrycounter
1474         \pageref{glentry-\glstetoklabel{#1}}%
1475       \else
1476         \gls{#1}%
1477       \fi
1478     \fi
1479   }
1480 }%

```

lossarypreamble

```

1481 \newcommand{\apptoglossarypreamble}[2][\glsdefaulttype]{%
1482   \ifcsdef{glolist@#1}%
1483   {%
1484     \ifcsundef{@glossarypreamble@#1}%
1485     {\csdef{@glossarypreamble@#1}{}}%
1486     {}%
1487     \csappto{@glossarypreamble@#1}{#2}%
1488   }%
1489   {%
1490     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
1491   }%
1492 }

```

lossarypreamble

```

1493 \newcommand{\preglossarypreamble}[2][\glsdefaulttype]{%
1494   \ifcsdef{glolist@#1}%
1495   {%
1496     \ifcsundef{@glossarypreamble@#1}%
1497     {\csdef{@glossarypreamble@#1}{}}%
1498     {}%
1499     \cspreto{@glossarypreamble@#1}{#2}%
1500   }%
1501   {%
1502     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
1503   }%
1504 }

```

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.

The original `\@gls@entry@field` causes a problem for undefined entries when used in section headings or captions. Since entries must be defined with just the base package this isn't a significant issue, but it will cause a problem with `bib2gls` where no entries are defined on the first \LaTeX call, so redefine `\@gls@entry@field` to use `\csuse` instead of `\csname`.

`\@gls@entry@field`

```
\@gls@entry@field{\<label>}{\<field>}
```

This command was introduced to glossaries version 4.03 but older versions are likely to be incompatible with glossaries-extra.

```
1505 \ifdef\@gls@entry@field
1506 {
1507   \renewcommand*{\@gls@entry@field}[2]{\csuse{glo@\glsdetoklabel{#1}@#2}}
1508 }
1509 {}
```

`\ifglsused`

```
\ifglsused{\<label>}{\<true part>}{\<false part>}
```

In the event that undefined entries should trigger a warning rather than an error, `\ifglsused` needs to be modified to check for existence. If the boolean variable is undefined, then its state is indeterminate and is neither true nor false, so neither *<true part>* nor *<false>* part will be performed if *<label>* is undefined. See also `\GlsXtrIfUnusedOrUndefined`.

```
1510 \renewcommand*{\ifglsused}[3]{%
1511   \glsdoifexists{#1}{\ifbool{glo@\glsdetoklabel{#1}@flag}{#2}{#3}}%
1512 }
```

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

`\longnewglossaryentry`

```
1513 \renewcommand*{\longnewglossaryentry}{%
1514   \@ifstar{\glxtr@s@longnewglossaryentry}{\glxtr@longnewglossaryentry}
1515 }
```

`\longnewglossaryentry` Starred version.

```
1516 \newcommand{\glxtr@s@longnewglossaryentry}[3]{%
```

```

1517 \glsdoifnoexists{#1}%
1518 {%
1519   \bgroup
1520   \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1521   \long\def\@newglossaryentryprehook{%
1522     \long\def\@glo@desc{#3}%
1523     \@org@newglossaryentryprehook
1524   }%
1525   \renewcommand*{\gls@assign@desc}[1]{%
1526     \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
1527     \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
1528   }
1529   \gls@defglossaryentry{#1}{#2}%
1530 \egroup
1531 }%
1532 }

```

`ewglossaryentry` Unstarred version.

```

1533 \newcommand{\@glsxtr@longnewglossaryentry}[3]{%
1534   \glsdoifnoexists{#1}%
1535   {%
1536     \bgroup
1537     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1538     \long\def\@newglossaryentryprehook{%
1539       \long\def\@glo@desc{#3\glsxtrpostlongdescription}%
1540       \@org@newglossaryentryprehook
1541     }%
1542     \renewcommand*{\gls@assign@desc}[1]{%
1543       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%

```

The following is different from the base glossaries.sty:

```

1544   \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
1545   }
1546   \gls@defglossaryentry{#1}{#2}%
1547 \egroup
1548 }%
1549 }

```

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

```

1550 \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.

```

1551 \renewcommand{\newignoredglossary}{%
1552   \@ifstar\glsxtr@s@newignoredglossary\glsxtr@org@newignoredglossary
1553 }

```

ignoredglossary The original definition is patched to check for existence.

```
1554 \newcommand*{\glxtr@org@newignoredglossary}[1]{%
1555   \ifcsdef{glolist@#1}
1556   {%
1557     \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
1558   }%
1559   {%
1560     \ifdefempty\@ignored@glossaries
1561     {%
1562       \protected@edef\@ignored@glossaries{#1}%
1563     }%
1564     {%
1565       \protected@eappto\@ignored@glossaries{,#1}%
1566     }%
1567     \csgdef{glolist@#1}{,}%
1568     \ifcsundef{gls@#1@entryfmt}%
1569     {%
1570       \defglsentryfmt[#1]{\glsentryfmt}%
1571     }%
1572     {}%
1573     \ifdefempty\@gls@nohyperlist
1574     {%
1575       \renewcommand*{\@gls@nohyperlist}{#1}%
1576     }%
1577     {%
1578       \protected@eappto\@gls@nohyperlist{,#1}%
1579     }%
1580   }%
1581 }
```

ignoredglossary Starred form.

```
1582 \newcommand*{\glxtr@s@newignoredglossary}[1]{%
1583   \ifcsdef{glolist@#1}
1584   {%
1585     \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
1586   }%
1587   {%
1588     \ifdefempty\@ignored@glossaries
1589     {%
1590       \protected@edef\@ignored@glossaries{#1}%
1591     }%
1592     {%
1593       \protected@eappto\@ignored@glossaries{,#1}%
1594     }%
1595     \csgdef{glolist@#1}{,}%
1596     \ifcsundef{gls@#1@entryfmt}%
1597     {%
```

```

1598     \defglentryfmt[#1]{\glentryfmt}%
1599   }%
1600   {}%
1601 }%
1602 }

```

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

```

1603 \glusifusetranslator
1604 {%
1605   \renewcommand*{\glsettoctitle}[1]{%
1606     \ifcsdef{glstr@set@#1@toctitle}%
1607       {%
1608         \csuse{glstr@set@#1@toctitle}%
1609       }%
1610     {%
1611       \ifcsdef{@glotype@#1@title}%
1612         {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1613         {\def\glossarytoctitle{\glossarytitle}}%
1614       }%
1615     }%
1616 }
1617 {
1618   \renewcommand*{\glsettoctitle}[1]{%
1619     \ifcsdef{@glotype@#1@title}%
1620       {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1621       {\def\glossarytoctitle{\glossarytitle}}%
1622   }
1623 }

```

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

```

1624 \newcommand{\provideignoredglossary}{%
1625   \@ifstar\glstr@s@provideignoredglossary\glstr@provideignoredglossary
1626 }

```

`ignoredglossary` Unstarred version.

```

1627 \newcommand*{\glstr@provideignoredglossary}[1]{%
1628   \ifcsdef{glolist@#1}
1629     {}%
1630     {%
1631       \ifdefempty\@ignored@glossaries
1632         {%
1633           \protected@edef\@ignored@glossaries{#1}%
1634         }%
1635         {%
1636           \protected@eappto\@ignored@glossaries{,#1}%
1637         }%
1638       \csgdef{glolist@#1}{,}%

```

```

1639 \ifcsundef{gls@#1@entryfmt}%
1640 {%
1641 \defglsentryfmt[#1]{\glsentryfmt}%
1642 }%
1643 {}%
1644 \ifdefempty\@gls@nohyperlist
1645 {%
1646 \renewcommand*{\@gls@nohyperlist}{#1}%
1647 }%
1648 {%

1649 \protected@eappto\@gls@nohyperlist{, #1}%
1650 }%
1651 }%
1652 }

```

`ignoredglossary` Starred form.

```

1653 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
1654 \ifcsdef{glolist@#1}
1655 {}%
1656 {%

1657 \ifdefempty\@ignored@glossaries
1658 {%
1659 \protected@edef\@ignored@glossaries{#1}%
1660 }%
1661 {%
1662 \protected@eappto\@ignored@glossaries{, #1}%
1663 }%
1664 \csgdef{glolist@#1}{,}%
1665 \ifcsundef{gls@#1@entryfmt}%
1666 {%
1667 \defglsentryfmt[#1]{\glsentryfmt}%
1668 }%
1669 {}%
1670 }%
1671 }

```

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

```

1672 \newcommand*{\glxtrcopytoglossary}[2]{%
1673 \glsdoifexists{#1}%
1674 {%
1675 \ifcsdef{glolist@#2}
1676 {%

1677 \protected@cseappto{glolist@#2}{#1,}%
1678 }%
1679 {%
1680 \glxtrundefaction{Glossary type ‘#2’ doesn’t exist}{}%

```



```

1681 }%
1682 }%
1683 }

```

1.3.1 Existence Checks

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

```

1684 \renewcommand{\glsdoifexists}[2]{%
1685   \ifglentryexists{#1}{#2}%
1686   {%

```

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

```

1687   \protected@edef\glslabel{\glsdetoklabel{#1}}%
1688   \glstrundefaction{Glossary entry '\glslabel'
1689     has not been defined}{You need to define a glossary entry before
1690     you can reference it.}%
1691   }%
1692 }

```

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

```

1693 \renewcommand{\glsdoifnoexists}[2]{%
1694   \ifglentryexists{#1}{%
1695     \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
1696       has already been defined}{}}{#2}%
1697 }

```

`\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.

```

1698 \ifdef\glsdoifexistsordo
1699 {%
1700   \renewcommand{\glsdoifexistsordo}[3]{%
1701     \ifglentryexists{#1}{#2}%
1702     {%
1703       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
1704         has not been defined}{You need to define a glossary entry
1705         before you can use it.}%
1706       #3%
1707     }%
1708   }%
1709 }
1710 {%
1711   \glstr@warnonexistsordo\glsdoifexistsordo
1712   \newcommand{\glsdoifexistsordo}[3]{%
1713     \ifglentryexists{#1}{#2}%
1714     {%
1715       \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
1716         has not been defined}{You need to define a glossary entry
1717         before you can use it.}%

```

```

1718      #3%
1719    }%
1720  }%
1721 }

```

arynoexistsordo Similarly for \doifglossarynoexistsordo.

```

1722 \ifdef\doifglossarynoexistsordo
1723 {%
1724   \renewcommand{\doifglossarynoexistsordo}[3]{%
1725     \ifglossaryexists*{#1}%
1726     {%
1727       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1728       #3%
1729     }%
1730     {#2}%
1731   }%
1732 }
1733 {%
1734   \glstr@warnonexistsordo\doifglossarynoexistsordo
1735   \newcommand{\doifglossarynoexistsordo}[3]{%
1736     \ifglossaryexists*{#1}%
1737     {%
1738       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1739       #3%
1740     }%
1741     {#2}%
1742   }%
1743 }
1744

```

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.

```

1745 \appto\newglossaryentryposthook{%
1746   \ifdefvoid\@glo@see
1747   {\csxdef{glo@\@glo@label @see}{}}%
1748   {%
1749     \csxdef{glo@\@glo@label @see}{\@glo@see}%
1750     \if@glstr@autoseeindex
1751       \@glstr@autoindexcrossrefs
1752     \fi
1753   }%
1754 }
1755 \appto\@gls@keymap{,{see}{see}}

```

\glstrusesee Apply \glseeformat to the see key if not empty.

```

1756 \newcommand*{\glstrusesee}[1]{%

```

```

1757 \glsdoifexists{#1}%
1758 {%
1759   \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@see}%
1760   \ifdefempty\@glo@see
1761     {}%
1762     {%
1763       \expandafter\glsxtr@usesee\@glo@see\@end@glsxtr@usesee
1764     }%
1765   }%
1766 }

```

`\glsxtr@usesee`

```

1767 \newcommand*{\glsxtr@usesee}[1][\seename]{%
1768   \@glsxtr@usesee{#1}%
1769 }

```

`\@glsxtr@usesee`

```

1770 \def\@glsxtr@usesee[#1]#2\@end@glsxtr@usesee{%
1771   \glsxtruseseeformat{#1}{#2}%
1772 }

```

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

```

1773 \newcommand*{\glsxtruseseeformat}[2]{%
1774   \glsseeformat{#1}{#2}{}%
1775 }

```

`lsseeitemformat` glossaries originally defined `\glsseeitemformat` to use `\glsentryname` but in v3.0 this was switched to use `\glsentrytext` due to problems occurring with the name field being sanitized. Since this is no longer a problem, glossaries-extra restored the original definition as it makes more sense to use the name in the cross-reference list. Unfortunately this doesn't take style changes into account, so as from v1.42, this now uses `\glsfmttext` and `\glsfmtname` instead. (The text field is chosen rather than the short field to allow for the “noshort” styles.)

```

1776 \renewcommand*{\glsseeitemformat}[1]{%
1777   \ifglshashshort{#1}{\glsfmttext{#1}}{\glsfmtname{#1}}%
1778 }

```

`\glsxtrhiername`

`\glsxtrhiername{<label>}`

Displays the hierarchical name for the given entry. The cross-reference format `\glsseeitemformat` may be redefined to use this command to show the hierarchy, if required. This now uses `\glsfmttext` and `\glsfmtname` instead of `\glsaccessshort` and `\glsaccessname` to allow for style formatting.

```

1779 \newcommand*{\glsxtrhiername}[1]{%

```

```

1780 \glsdoifexists{#1}%
1781 {%
1782   \glstrifhasfield{parent}{#1}%
1783   {\glstrhiername{\glscurrentfieldvalue}\glstrhiernamesep}%
1784   }%
1785   \ifglshasshort{#1}{\glsfmtext{#1}}{\glsfmname{#1}}%
1786 }%
1787 }

```

\Glsxtrhiername

`\Glsxtrhiername{<label>}`

As above but displays the top-level name with an initial capital.

```

1788 \newcommand*{\Glsxtrhiername}[1]{%
1789   \glsdoifexists{#1}%
1790   {%
1791     \glstrifhasfield{parent}{#1}%
1792     {%
1793       \Glsxtrhiername{\glscurrentfieldvalue}\glstrhiernamesep
1794       \ifglshasshort{#1}{\glsfmtext{#1}}{\glsfmname{#1}}%
1795     }%
1796     {\ifglshasshort{#1}{\Glsfmtext{#1}}{\Glsfmname{#1}}}%
1797   }%
1798 }

```

\GlsXtrhiername

`\GlsXtrhiername{<label>}`

As above but converts the first letter of each name to a capital. (Note that this isn't applying title case, just capitalising the start of each hierarchical element.)

```

1799 \newcommand*{\GlsXtrhiername}[1]{%
1800   \glsdoifexists{#1}%
1801   {%
1802     \glstrifhasfield{parent}{#1}%
1803     {\GlsXtrhiername{\glscurrentfieldvalue}\glstrhiernamesep}%
1804     }%
1805     \ifglshasshort{#1}{\Glsfmtext{#1}}{\Glsfmname{#1}}%
1806   }%
1807 }

```

\GLSxtrhiername

$\backslash\text{GLSxtrhiername}\{\langle label \rangle\}$

As above but displays the top-level name in all-caps.

```

1808 \newcommand*{\GLSxtrhiername}[1]{%
1809   \glsdoifexists{#1}%
1810   {%
1811     \glsxtrifhasfield{parent}{#1}%
1812     {%
1813       \GLSxtrhiername{\glscurrentfieldvalue}\glsxtrhiernamesep
1814       \ifglshasshort{#1}{\glsfmttext{#1}}{\glsfmtname{#1}}%
1815     }%
1816     {\ifglshasshort{#1}{\GLSfmttext{#1}}{\GLSfmtname{#1}}}%
1817   }%
1818 }
```

$\backslash\text{GLSXTRhiername}$

$\backslash\text{GLSXTRhiername}\{\langle label \rangle\}$

As above but displays all names in all-caps.

```

1819 \newcommand*{\GLSXTRhiername}[1]{%
1820   \glsdoifexists{#1}%
1821   {%
1822     \glsxtrifhasfield{parent}{#1}%
1823     {\GLSXTRhiername{\glscurrentfieldvalue}\glsxtrhiernamesep}%
1824     {}
1825     \ifglshasshort{#1}{\GLSfmttext{#1}}{\GLSfmtname{#1}}%
1826   }%
1827 }
```

sxtrhiernamesep Separator used in $\backslash\text{glsxtrhiername}$ and variants.

```

1828 \newcommand*{\glsxtrhiernamesep}{\,\,\small$\triangleright$}\,}
```

lsxtruseseealso Apply $\backslash\text{glsseeformat}$ to the `seealso` key if not empty. There's no optional tag to worry about here.

```

1829 \newcommand*{\glsxtruseseealso}[1]{%
1830   \glsdoifexists{#1}%
1831   {%
1832     \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@seealso}%
1833     \ifdefempty\@glo@see
1834     {}%
1835     {%
1836       \expandafter\glsxtruseseealsoformat\expandafter{\@glo@see}%
1837     }%
1838   }%
1839 }
```

`\glxtrusealias` Apply `\glssseeformat` to the alias key if not empty. There's no optional tag to worry about here. The value also isn't a comma-separated list, but use the same interface.

```
1840 \newcommand*{\glxtrusealias}[1]{%
1841   \glsdoidexists{#1}%
1842   {%
1843     \letcs{\@glo@see}{glo\glsdetoklabel{#1}@alias}%
1844     \ifdefempty\@glo@see
1845       {}%
1846       {%
```

Expansion isn't necessary because the value is a single label not a list.

```
1847     \glxtrusesseeformat{\seenname}{\@glo@see}%
1848   }%
1849 }%
1850 }
```

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

```
1851 \newcommand*{\glxtrusessealsoformat}[1]{%
1852   \glssseeformat[\seealso]{#1}%
1853 }
```

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

```
1854 \newrobustcmd{\glxtrseelist}[1]{%
1855   \protected@edef\@glo@tmp{\noexpand\glssseelist{#1}}\@glo@tmp
1856 }
```

`\glssseelist` Redefine to make `\glssseelist` more flexible.

```
1857 \renewrobustcmd*{\glssseelist}[1]{%
1858   \let\@gls@dolast\relax
1859   \let\@gls@donext\relax
1860   \let\@glsseeitem\@glxtr@seefirstitem
1861   \let\@glsseelastsep\glssseelastsep
1862   \@for\@gls@thislabel:=#1\do{%
1863     \ifx\@xfor@nextelement\@nnil
1864       \@gls@dolast
1865     \else
1866       \@gls@donext
1867     \fi
1868     \expandafter\@glsseeitem\expandafter{\@gls@thislabel}%
1869     \let\@gls@dolast\@glsseelastsep
1870     \let\@gls@donext\glssseesep
1871     \let\@glsseeitem\@glxtr@seeitem
1872     \let\@glsseelastsep\glssseelastoxfordsep
1873   }%
1874 }
```

@glxtr@seeitem

```
1875 \newcommand*{\@glxtr@seeitem}[1]{%
1876   \glxtrifmulti{#1}{\mglseeitem{#1}}{\glseeitem{#1}}%
1877 }
```

tr@seefirstitem

```
1878 \newcommand*{\@glxtr@seefirstitem}[1]{%
1879   \glxtrifmulti{#1}{\mglseeefirstitem{#1}}{\glseeefirstitem{#1}}%
1880 }
```

\mglseeitem Multi-entry cross-reference

```
1881 \newcommand*{\mglseeitem}[1]{%
1882   \mglename[all={noindex},setup={hyper=allmain}]{#1}%
1883 }
```

glseeefirstitem Multi-entry cross-reference

```
1884 \newcommand*{\mglseeefirstitem}{\mglseeitem}
```

glseeefirstitem

```
1885 \newcommand*{\glseeefirstitem}{\glseeitem}
```

eeelastoxfordsep

```
1886 \newcommand*{\glseeelastoxfordsep}{\glseeelastsep}
```

\seealso In case this command hasn't been defined. Languages packages actually provide \alsoname so use that if it's defined.

```
1887 \ifdef\alsoname
1888 {\providecommand\seealso}{\alsoname}}
1889 {\providecommand\seealso}{see also}}
```

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

```
1890 \ifdef\@xdycrossrefhook
1891 {
  Add the cross-reference class definition to the hook.
1892   \appto\@xdycrossrefhook{%
1893     \write\glswrite{(define-crossref-class \string"seealso\string"
1894       :unverified )}%
1895     \write\glswrite{(markup-crossref-list
1896       :class \string"seealso\string"^^J\space\space\space
1897       :open \string"\string\glxtruseealsoformat\glsoopenbrace\string"
1898       :close \string"\glsclosebrace\string")}%
1899   }
```

Append to class list.

```
1900   \appto\@xdylocationclassorder{\space\string"seealso\string"}
```

This essentially works like `\@do@seeglossary` but uses the `seealso` class. This doesn't increment the associated counter.

```

1901 \newrobustcmd*{\glxtrindexseealso}[2]{%
1902   \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
1903     \@glxtr@recordsee{#1}{#2}%
1904   \fi
1905   \glsoifexists{#1}%
1906   {%
1907     \@glxtrwrglossmark
1908     \def\@gls@xref{#2}%
1909     \@onelevel@sanitize\@gls@xref
1910     \@gls@checkmkidxchars\@gls@xref
1911     \gls@glossary{\csname glo@#1@type\endcsname}{%
1912       (indexentry
1913         :tkey (\csname glo@#1@index\endcsname)
1914         :xref (\string"\@gls@xref\string")
1915         :attr \string"seealso\string"
1916       )
1917     }%
1918   }%
1919 }
1920 }
1921 {
  xindy not in use or glossaries version too old to support this.
1922 \newrobustcmd*{\glxtrindexseealso}{\glssee[\seealso]}
1923 }

```

The `alias` key should be set to the label of the synonymous entry. The `seealso` key essentially behaves like `see=[\seealso]{\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`.

```

1924 \ifdef\gls@set@xr@key
1925 {

```

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

```

1926 \define@key{glossentry}{alias}{%
1927   \gls@set@xr@key{alias}{\@glo@alias}{#1}%
1928 }
1929 \define@key{glossentry}{seealso}{%
1930   \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
1931 }

```

Add to the key mappings.

```

1932 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}

```

Set the default value.

```

1933 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%

```


Assign the field values.

```

1934 \appto\@newglossaryentryposthook{%
1935   \ifdefvoid\@glo@seealso
1936     {\csxdef{glo@\@glo@label @seealso}{}}%
1937     {%
1938       \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1939       \if@glxtr@autoseeindex
1940         \@glxtr@autoindexcrossrefs
1941       \fi
1942     }%

```

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

```

1943   \ifdefvoid\@glo@alias
1944     {\csxdef{glo@\@glo@label @alias}{}}%
1945     {%
1946       \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1947     }%
1948 }

```

Provide user-level commands to access the values.

\glxtralias

```

1949 \newcommand*\glxtralias[1]{\@gls@entry@field{#1}{alias}}

```

trseealsolabels

```

1950 \newcommand*\glxtrseealsolabels[1]{\@gls@entry@field{#1}{seealso}}

```

Add to the \@glo@autosee hook.

```

1951 \appto\@glo@autoseehook{%
1952   \ifdefvoid\@glo@alias
1953     {%
1954       \ifdefvoid\@glo@seealso
1955         {}%
1956       {%
1957         \protected@edef\@do@glsee{\noexpand\glxtrindexseealso
1958           {\@glo@label}{\@glo@seealso}}%
1959         \@do@glsee
1960       }%
1961     }%
1962   {%

```

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

```

1963     \ifdefvoid\@glo@see
1964     {%
1965       \protected@edef\@do@glsee{\noexpand\glsee{\@glo@label}{\@glo@alias}}%
1966       \@do@glsee
1967     }%
1968     {}%

```

```

1969 }%
1970 }%
1971 }
1972 {

```

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

`\glsxtralias`

```

1973 \glsaddstoragekey*{alias}{\glsxtralias}

```

`trseealsolabels`

```

1974 \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.

```

1975 \appto\@newglossaryentryposthook{%
1976 \ifcsvoid{glo@\@glo@label @alias}%
1977 {%
1978 \ifcsvoid{glo@\@glo@label @seealso}%
1979 }%
1980 {%

1981 \protected@edef\@do@glsssee{\noexpand\glsxtrindexseealso
1982 {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1983 \@do@glsssee
1984 }%
1985 }%
1986 {%

```

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

```

1987 \ifdefvoid\@glo@see
1988 {%

1989 \protected@edef\@do@glsssee{\noexpand\glsssee
1990 {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1991 \@do@glsssee
1992 }%
1993 }%
1994 }%
1995 }

1996 }

```

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

```

1997 \AtEndDocument{\if@glsxtrindexcrossrefs\glsxtraddallcrossrefs\fi}

```

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

```

1998 \newcommand*{\glxtraddallcrossrefs}{%
1999   \forallglossaries{\@glo@type}%
2000   {%
2001     \forglsentries[\@glo@type]{\@glo@label}%
2002     {%
2003       \ifglsused{\@glo@label}%
2004       {\expandafter\@glxtr@addunusedxrefs\expandafter{\@glo@label}}}%
2005     }%
2006   }%
2007 }

```

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

```

2008 \newcommand*{\@glxtr@addunusedxrefs}[1]{%
2009   \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
2010   \ifdefvoid\@glo@see
2011   {}%
2012   {%
2013     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
2014   }%
2015   \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@seealso}%
2016   \ifdefvoid\@glo@see
2017   {}%
2018   {%
2019     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
2020   }%
2021 }

```

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

```

2022 \newcommand*{\glxtr@addunused}[1][ ]{%
2023   \@glxtr@addunused
2024 }

```

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

```

2025 \def\@glxtr@addunused#1\@end@glxtr@addunused{%
2026   \@for\@glxtr@label:=#1\do
2027   {%
2028     \glxtrifmulti\@glxtr@label
2029     {%
2030       \letcs\@glxtr@labellist{\@glsc@combined\@glxtr@label @list}%
2031       \@for\@glxtr@multilabel:=\@glxtr@labellist\do
2032       {\@glxtr@addunused\@glxtr@multilabel\@end@glxtr@addunused}%
2033     }%
2034     {%
2035       \ifglsused{\@glxtr@label}{}%
2036       {%
2037         \glsadd[format=glxtrunusedformat]{\@glxtr@label}%
2038         \glsunset{\@glxtr@label}%
2039         \expandafter\@glxtr@addunusedxrefs\expandafter{\@glxtr@label}%

```

```

2040     }%
2041 }%
2042 }%
2043 }

```

xtrunusedformat

```

2044 \newcommand*{\glxtrunusedformat}[1]{\unskip}

```

1.3.2 Document Definitions

`\ls@begindocdefs` This command was only introduced to glossaries v4.37, so it may not be defined. If it has been defined, redefine it to check `\@glxtr@docdefval` so that it only inputs the `.glsdefs` file if `docdef=true`.

```

2045 \ifdef\gl@begindocdefs
2046 {%
2047   \renewcommand*{\gl@begindocdefs}{%
2048     \ifnum\@glxtr@docdefval=1\relax
2049       \@gl@enablesavenonumberlist
2050       \edef\@gl@restoreat{%
2051         \noexpand\catcode'\noexpand\@=\number\catcode'\@ \relax}%
2052       \makeatletter
2053       \InputIfFileExists{\jobname.glsdefs}{\relax}%
2054       \@gl@restoreat
2055       \undef\@gl@restoreat
2056       \gl@defdocnewglossaryentry
2057     \else
2058       \ifnum\@glxtr@docdefval=3\relax

```

The `docdef=atom` package option has been set. Create the `.glsdefs` file for the autocomplete support but don't read it.

```

2059       \@gl@enablesavenonumberlist
2060       \let\gl@checkseeallowed\relax
2061       \let\newglossaryentry\new@atom@glossaryentry
2062       \global\newwrite\@gl@deffile
2063       \immediate\openout\@gl@deffile=\jobname.glsdefs

```

Write all currently defined entries.

```

2064       \forallgl@entries{\gl@entry}{\gl@writedef{\gl@entry}}%
2065     \fi
2066   \fi
2067 }
2068 }
2069 {%
2070   \ifnum\@glxtr@docdefval=3\relax
2071     \PackageError{glossaries-extra}{Package option
2072       'docdef=\@glxtr@docdefsetting' requires at least version 4.37
2073       of the base glossaries.sty package}{}
2074   \fi
2075 }

```

m@glossaryentry

```
2076 \newrobustcmd{\new@atom@glossaryentry}[2]{%
2077   \gls@defglossaryentry{#1}{#2}%
2078   \@gls@writedef{#1}%
2079 }
```

noidxglossaries Modify \makenoidxglossaries so that it automatically sets docdef=false (unless the restricted setting is on) and disables the docdef key. This command isn't allowed with the record option.

```
2080 \let\glsxtr@orgmakenoidxglossaries\makenoidxglossaries
2081 \renewcommand{\makenoidxglossaries}{%
2082   \@domakeglossaries
2083   {%
2084     \ifdefequal\@glsxtr@record@setting\@glsxtr@record@setting@off
2085     {%
2086       \glsxtr@orgmakenoidxglossaries
```

Add marker to \do@seeglossary but don't increment associated counter.

```
2087   \renewcommand{\do@seeglossary}[2]{%
2088     @@glsxtrwrglossmark

2089     \protected@edef\@gls@label{\glsdetoklabel{##1}}%
2090     \protected@write\@auxout{}{%
2091       \string\@gls@reference
2092       {\csname glo@\@gls@label @type\endcsname}%
2093       {\@gls@label}%
2094       {%
2095         \string\glsseeformat##2}%
2096       }%
2097     }%
2098   }%
```

Check for docdefs=restricted:

```
2099   \if@glsxtrdocdefrestricted
```

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

```
2100     \renewcommand*\@gls@reference}[3]{%
2101       \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
2102       \ifinlistcs{##2}{\glsref@##1}%
2103       {}%
2104       {\listcsgadd{\glsref@##1}{##2}}%
2105       \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
2106       {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}{}%
2107       {}%
2108       \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
2109     }%
2110   \else
```

Disable document definitions.

```

2111     \@glxtrdocdeffalse
2112     \fi
2113     \disable@keys{glossaries-extra}{\docdef}%
2114 }%
2115 {%
2116     \PackageError{glossaries-extra}{\string\makenoidxglossaries\space
2117     not permitted\MessageBreak
2118     with record=\@glxtr@record@setting\space package option}%
2119     {You may only use \string\makenoidxglossaries\ space with the
2120     record=off option}%
2121 }%
2122 }%
2123 }

```

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

```

2124 \renewcommand*{\gls@defdocnewglossaryentry}{%
2125     \ifcase\@glxtr@docdefval
2126         docdef=false:
2127         \renewcommand*{\newglossaryentry}[2]{%
2128             \PackageError{glossaries-extra}{Glossary entries must
2129             be \MessageBreak defined in the preamble with \MessageBreak
2130             package option ‘docdef=false’\MessageBreak(consider using
2131             ‘docdef=restricted’)}{Move your glossary definitions to
2132             the preamble. You can also put them in a \MessageBreak separate file
2133             and load them with \string\loadglsentries.}%
2134         }%
2135     \or

```

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

```

2135     \let\gls@checkseeallowed\relax
2136     \let\newglossaryentry\newglossaryentry
2137 \else

```

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

```

2138     \let\gls@checkseeallowed\relax
2139     \fi
2140 }%

```

Permit a special form of document definition, but only allow it if the glossaries come at the end of the document. These commands behave a little like a combination of `\newterm` and `\gls`. This must be explicitly enabled with the following.

`\rEnableOnTheFly`

```

2141 \newcommand*{\GlsXtrEnableOnTheFly}{%
2142     \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
2143 }

```

`\GlsXtrEnableOnTheFly` 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.)

```

2144 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
2145   \renewcommand*{\glsdetoklabel}[1]{%
2146     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
2147     {%
2148       \expandafter\detokenize\expandafter{##1}%
2149     }%
2150     {\detokenize{##1}}}%
2151   }%
2152   \@GlsXtrEnableOnTheFly
2153 }
2154 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
2155   \expandafter\if\glsbackslash#1%
2156   #3%
2157   \else
2158   #4%
2159   \fi
2160 }

```

`\GlsXtrStarflywarn`

```

2161 \newcommand*{\GlsXtrStarflywarn}{%
2162   \GlossariesExtraWarning{Experimental starred version of
2163   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
2164   read the warnings in the glossaries-extra user manual)}}%
2165 }

```

`\GlsXtrEnableOnTheFly`

```

2166 \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`

```

2167   \newcommand*{\glsxtrcat}{general}

```

`\glsxtr`

```

2168   \newcommand*{\glsxtr}[1] [] {%
2169     \def\glsxtr@keylist{##1}%
2170     \@glsxtr
2171   }

```

`\@glsxtr`

```

2172   \newcommand*{\@glsxtr}[2] [] {%
2173     \ifglsentryexists{##2}%

```

```

2174   {%
2175     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
2176   }%
2177   {%
2178     \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
2179       description={\nopostdesc},##1}%
2180   }%
2181   \expandafter\gls\expandafter[\glstr@keylist]{##2}%
2182 }

```

\Glsxtr

```

2183 \newcommand*{\Glsxtr}[1] [] {%
2184   \def\glstr@keylist{##1}%
2185   \@Glsxtr
2186 }

```

\@Glsxtr

```

2187 \newcommand*{\@Glsxtr}[2] [] {%
2188   \ifglstryexists{##2}%
2189   {%
2190     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
2191   }%
2192   {%
2193     \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
2194       description={\nopostdesc},##1}%
2195   }%
2196   \expandafter\gls\expandafter[\glstr@keylist]{##2}%
2197 }

```

\glstrpl

```

2198 \newcommand*{\glstrpl}[1] [] {%
2199   \def\glstr@keylist{##1}%
2200   \@glstrpl
2201 }

```

\@glstrpl

```

2202 \newcommand*{\@glstrpl}[2] [] {%
2203   \ifglstryexists{##2}%
2204   {%
2205     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
2206   }%
2207   {%
2208     \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
2209       description={\nopostdesc},##1}%
2210   }%
2211   \expandafter\glsp\expandafter[\glstr@keylist]{##2}%
2212 }

```

\Glsxtrpl


```

2213 \newcommand*{\Glsxtrpl}[1][]{%
2214 \def\glxtr@keylist{##1}%
2215 \@Glsxtrpl
2216 }

```

\@Glsxtrpl

```

2217 \newcommand*{\@Glsxtrpl}[2][]{%
2218 \ifglstryentryexists{##2}
2219 {%
2220 \ifblank{##1}{\GlsXtrWarning{##1}{##2}}%
2221 }%
2222 {%
2223 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
2224 description={\nopostdesc},##1}%
2225 }%
2226 \expandafter\Glspl\expandafter[\glxtr@keylist]{##2}%
2227 }

```

\GlsXtrWarning

```

2228 \newcommand*{\GlsXtrWarning}[2]{%
2229 \def\glxtr@optlist{##1}%
2230 \@onelevel@sanitize\glxtr@optlist
2231 \GlossariesExtraWarning{The options '\glxtr@optlist' have
2232 been ignored for entry '##2' as it has already been defined}%
2233 }

```

Disable commands after the glossary:

```

2234 \renewcommand\@printglossary[2]{%
2235 \def\glxtr@printglossopts{##1}%
2236 \@glxtr@orgprintglossary{##1}{##2}%
2237 \def\glxtr{\glxtr@disabledflycommand\glxtr}%
2238 \def\glxtrpl{\glxtr@disabledflycommand\glxtrpl}%
2239 \def\@Glsxtr{\glxtr@disabledflycommand\Glsxtr}%
2240 \def\@Glsxtrpl{\glxtr@disabledflycommand\Glsxtrpl}%
2241 }

```

abledflycommand

```

2242 \newcommand*{\@glxtr@disabledflycommand}[1]{%
2243 \PackageError{glossaries-extra}%
2244 {\string##1\space can't be used after any of the \MessageBreak
2245 glossaries have been displayed}%
2246 {The on-the-fly commands enabled by
2247 \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
2248 before the glossaries. If you want to use any entries \MessageBreak
2249 after any of the glossaries, you must use the standard \MessageBreak
2250 method of first defining the entry and then using the \MessageBreak
2251 entry with commands like \string\gls}%
2252 \@glxtr@disabledflycommand
2253 }%

```

```
2254 \newcommand*{\@glxtr@disabledflycommand}[2][\@##2]
```

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

```
2255 \let\GlsXtrEnableOnTheFly\relax
2256 }
2257 \@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.

```
2258 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}
```

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

etglossarystyle

```
2259 \renewcommand*{\setglossarystyle}[1]{%
2260   \ifcsundef{@glsstyle@#1}%
2261   {%
2262     \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%
2263   }%
2264   {%
2265     \csname @glsstyle@#1\endcsname
```

Only set the current style if it exists.

```
2266   \protected@edef\@glxtr@current@style{#1}%
2267   }%
2268   \ifx\@glossary@default@style\relax
2269   \protected@edef\@glossary@default@style{#1}%
2270   \fi
2271 }
```

In case we have an old version of glossaries:

```
2272 \ifdef\@glossary@default@style
2273 {}
2274 {%
2275   \let\@glossary@default@style\relax
2276 }
```

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

```
2277 \ifdef\glslistdottedwidth
2278 {%
2279   \ifdim\glslistdottedwidth=.5\hsize
2280     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
2281     \AtBeginDocument{%
2282       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
```

```

2283     \setlength{\glslistdottedwidth}{.5\columnwidth}%
2284     \fi
2285   }%
2286 \fi
2287 }
2288 {}%

```

Similarly for \glsdescwidth:

\glsdescwidth

```

2289 \ifdef\glsdescwidth
2290 {%
2291   \ifdim\glsdescwidth=.6\hsize
2292     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
2293     \AtBeginDocument{%
2294       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
2295         \setlength{\glsdescwidth}{.6\columnwidth}%
2296       \fi
2297     }%
2298   \fi
2299 }
2300 {}%

```

and for \glspagelistwidth:

\glspagelistwidth

```

2301 \ifdef\glspagelistwidth
2302 {%
2303   \ifdim\glspagelistwidth=.1\hsize
2304     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
2305     \AtBeginDocument{%
2306       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
2307         \setlength{\glspagelistwidth}{.1\columnwidth}%
2308       \fi
2309     }%
2310   \fi
2311 }
2312 {}%

```

aryentrynumbers Has the nonnumberlist option been used?

```

2313 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
2314 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
2315   \glsnonnumberlistfalse
2316   \renewcommand*{\glossaryentrynumbers}[1]{%
2317     \ifglstryentryexists{\glscurrententrylabel}%
2318     {%
2319       \@glsxtrpreloctag
2320       \GlsXtrFormatLocationList{#1}%
2321       \@glsxtrpostloctag
2322       \gls@save@numberlist{#1}%

```

```

2323     }{}%
2324 }%
2325 \else
2326   \glsnonnumberlisttrue
2327   \renewcommand*{\glossaryentrynumbers}[1]{%
2328     \ifglsentryexists{\glscurrententrylabel}%
2329     {%
2330       \gls@save@numberlist{#1}%
2331     }{}%
2332 }%
2333 \fi

```

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

```

2334 \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

```

2335 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
2336   \let\@glstrpreloctag\@glstrpreloctag
2337   \let\@glstrpostloctag\@glstrpostloctag
2338   \renewcommand*{\@glstr@pagetag}{#1}%
2339   \renewcommand*{\@glstr@pagetag}{#2}%
2340   \renewcommand*{\@glstr@savepreloctag}[2]{%
2341     \csgdef{\@glstr@preloctag@##1}{##2}%
2342   }%
2343   \renewcommand*{\@glstr@doloctag}{%
2344     \ifcsundef{\@glstr@preloctag\glscurrententrylabel}%
2345     {%
2346       \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
2347         Rerun required}%
2348     }%
2349     {%
2350       \csuse{\@glstr@preloctag\glscurrententrylabel}%
2351     }%
2352   }%
2353 }
2354 \@onlypreamble\GlsXtrEnablePreLocationTag

```

glstrpreloctag

```

2355 \newcommand*{\@glstrpreloctag}{%
2356   \let\@glstr@org@delimN\delimN
2357   \let\@glstr@org@delimR\delimR
2358   \let\@glstr@org@glsgignore\glsgignore

```

\gdef is required as the delimiters may occur inside a scope.

```

2359 \gdef\@glxtr@thisloctag{\@glxtr@pagetag}%
2360 \renewcommand*{\delimN}{\%
2361 \gdef\@glxtr@thisloctag{\@glxtr@pagetag}%
2362 \@glxtr@org@delimN}%
2363 \renewcommand*{\delimR}{\%
2364 \gdef\@glxtr@thisloctag{\@glxtr@pagetag}%
2365 \@glxtr@org@delimR}%
2366 \renewcommand*{\glsgignore}[1]{\%
2367 \gdef\@glxtr@thisloctag{\relax}%
2368 \@glxtr@org@glsgignore{##1}}%
2369 \@glxtr@doloctag
2370 }

```

glxtrpreloctag

```

2371 \newcommand*{\@glxtrpreloctag}{}

```

@glxtr@pagetag

```

2372 \newcommand*{\@glxtr@pagetag}{}%

```

glxtr@pagetag

```

2373 \newcommand*{\@glxtr@pagetag}{}%

```

lsxtrpostloctag

```

2374 \newcommand*{\@@glxtrpostloctag}{\%
2375 \let\delimN\@glxtr@org@delimN
2376 \let\delimR\@glxtr@org@delimR
2377 \let\glsgignore\@glxtr@org@glsgignore
2378 \protected@write\@auxout{}{\%
2379 {\string\@glxtr@savepreloctag{\glscurrententrylabel}\@glxtr@thisloctag}}%
2380 }

```

lsxtrpostloctag

```

2381 \newcommand*{\@glxtrpostloctag}{}

```

lsxtr@preloctag

```

2382 \newcommand*{\@glxtr@savepreloctag}[2]{\%
2383 \protected@write\@auxout{}{\%
2384 \string\providecommand\string\@glxtr@savepreloctag[2]{}}

```

glxtr@doloctag

```

2385 \newcommand*{\@glxtr@doloctag}{}

```

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

```

2386 \renewcommand*{\KV@printgloss@nonumberlist}[1]{\%
2387 \XKV@plfalse
2388 \XKV@sttrue

```

```

2389 \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
2390 {%
2391   \csname glsnonumberlist\XKV@resa\endcsname
2392   \ifglsnonumberlist
2393     \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
2394   \else
2395     \def\glossaryentrynumbers##1{%
2396       \@glstrpreloctag
2397       \GlsXtrFormatLocationList{##1}%
2398       \@glstrpostloctag
2399       \gls@save@numberlist{##1}}%
2400   \fi
2401 }%
2402 }

```

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.

```

2403 \renewcommand*{\glsentryfmt}{%
2404   \ifglshashshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}}%
2405   \glsifregular{\glslabel}%
2406   {\glstrregularfont{\glsgenentryfmt}}%
2407   {%
2408     \ifglshashshort{\glslabel}%
2409     {\glstrabbreviationfont{\glstrgenabbrvfmt}}%
2410     {\glstrregularfont{\glsgenentryfmt}}%
2411   }%
2412 }

```

glstrregularfont Font used for regular entries.

```
2413 \newcommand*{\glstrregularfont}[1]{#1}
```

glstrabbreviationfont Font used for abbreviation entries.

```
2414 \newcommand*{\glstrabbreviationfont}[1]{#1}
```

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

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

```
2415 \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 entry exists (unless indexing has been switched off).

```
2416 \@glsxtr@record{#2}{#3}{glslink}%
2417 \glsdoifexists{#3}%
2418 {%
```

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).

```
2419 \let\glsxtrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
```

Save local setting.

```
2420 \@gls@save@glslocal
2421 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2422 \def\glscustomtext{#4}%
2423 \@glsxtr@field@linkdefs
2424 #1%
2425 \@gls@link[#2]{#3}{#4}%
2426 \let\ifKV@glslink@hyper\glsxtrorg@ifKV@glslink@hyper
2427 \@gls@restore@glslocal
2428 }%
2429 \glspostlinkhook
2430 }
```

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

\@gls@ Save the original definition and redefine.

```
2431 \let\@glsxtr@org@gls@\@gls@
2432 \def\@gls@#1#2{%
2433 \@glsxtr@record{#1}{#2}{glslink}%
2434 \@glsxtr@org@gls@{#1}{#2}%
2435 }%
```

\@glspl@ Save the original definition and redefine.

```
2436 \let\@glsxtr@org@glspl@\@glspl@
2437 \def\@glspl@#1#2{%
2438 \@glsxtr@record{#1}{#2}{glslink}%
2439 \@glsxtr@org@glspl@{#1}{#2}%
2440 }%
```

\@Gls@ Save the original definition and redefine.

```
2441 \let\@glsxtr@org@Gls@\@Gls@
2442 \def\@Gls@#1#2{%
2443 \@glsxtr@record{#1}{#2}{glslink}%
2444 \@glsxtr@org@Gls@{#1}{#2}%
2445 }%
```

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

```
2446 \let\@glxtr@org@Glspl@\@Glspl@
2447 \def\@Glspl@#1#2{%
2448   \@glxtr@record{#1}{#2}{glslink}%
2449   \@glxtr@org@Glspl@{#1}{#2}%
2450 }%
```

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

```
2451 \let\@glxtr@org@GLS@\@GLS@
2452 \def\@GLS@#1#2{%
2453   \@glxtr@record{#1}{#2}{glslink}%
2454   \@glxtr@org@GLS@{#1}{#2}%
2455 }%
```

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

```
2456 \let\@glxtr@org@GLSpl@\@GLSpl@
2457 \def\@GLSpl@#1#2{%
2458   \@glxtr@record{#1}{#2}{glslink}%
2459   \@glxtr@org@GLSpl@{#1}{#2}%
2460 }%
```

`\@glsdisp` This is redefined to allow the recording on the first run. Can't save and restore `\@glsdisp` since it has an optional argument.

```
2461 \renewcommand*{\@glsdisp}[3][{}]{%
2462   \@glxtr@record{#1}{#2}{glslink}%
2463   \glsdoifexists{#2}{%
2464     \let\do@gl@link@checkfirsthyper\@gl@link@checkfirsthyper
2465     \let\glsifplural\@secondoftwo
2466     \let\glscapscase\@firstofthree
2467     \def\glscustomtext{#3}%
2468     \def\glsinsert{}%
2469     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
2470     \@gl@link[#1]{#2}{\@glo@text}%
2471     \@gl@do@gl@unset{#2}%
2472   }%
2473   \glspostlinkhook
2474 }
```

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

```
2475 \renewcommand*{\@gl@link}[3][{}]{%
2476   \@glxtr@record{#1}{#2}{glslink}%
2477   \glsdoifexistsordo{#2}%
2478   {%
2479     \let\do@gl@link@checkfirsthyper\relax
```

Post-link hook commands need initialising.

```
2480   \def\glscustomtext{#3}%
2481   \@glxtr@field@linkdefs
2482   \@gl@link[#1]{#2}{#3}%
```



```

2483 }%
2484 {%
2485   \glstextformat{#3}%
2486 }%
2487 \glspostlinkhook
2488 }

```

gsxtrinitwrgloss Set the default if the wrgloss is omitted.

```

2489 \newcommand*{\gsxtrinitwrgloss}{%
2490   \glsifattribute{\glslabel}{wrgloss}{after}%
2491   {%
2492     \gsxtrinitwrglossbeforefalse
2493   }%
2494   {%
2495     \gsxtrinitwrglossbeforetrue
2496   }%
2497 }

```

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

```

2498 \newif\ifgsxtrinitwrglossbefore
2499 \gsxtrinitwrglossbeforetrue

```

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

```

2500 \define@choicekey{glslink}{wrgloss}%
2501 [\@gsxtr@wrglossval\@gsxtr@wrglossnr]%
2502 {before,after}%
2503 {%
2504   \ifcase\@gsxtr@wrglossnr\relax
2505     \gsxtrinitwrglossbeforetrue
2506   \or
2507     \gsxtrinitwrglossbeforefalse
2508   \fi
2509 }

2510 \define@key{glslink}{thevalue}{\def\@gsxtr@thevalue{#1}}

2511 \define@key{glslink}{theHvalue}{\def\@gsxtr@theHvalue{#1}}

```

gsxtr@hyperoutside Define a hyperoutside key to determine whether \hyperlink should be outside \glstextformat.

```

2512 \define@boolkey{glslink}[gsxtr@]{hyperoutside}[true]{}
2513 \gsxtr@hyperoutsidettrue

```

gsxtr@local@textformat Provide a key to locally change the text format.

```

2514 \define@key{glslink}{textformat}{%
2515   \ifcsdef{#1}
2516   {%
2517     \letcs{\@gsxtr@local@textformat}{#1}%
2518   }%

```

```

2519  {%
2520    \PackageError{glossaries-extra}{Unknown control sequence name ‘#1’}{}%
2521  }%
2522 }

2523 \define@key{glslink}{prefix}{\def\glolinkprefix{#1}}

nithyperoutside  Set the default if the hyperoutside is omitted.
2524 \newcommand*{\glxtrnithyperoutside}{%
2525   \glsifattribute{\glslabel}{hyperoutside}{false}%
2526   {%
2527     \glxtr@hyperoutsidefalse
2528   }%
2529   {%
2530     \glxtr@hyperoutsidetrue
2531   }%
2532 }

r@inc@linkcount  Does nothing by default.
2533 \newcommand*{\glxtr@inc@linkcount}{}

slinkpresetkeys  User hook performed immediately before options are set. Does nothing by default.
2534 \newcommand*{\glslinkpresetkeys}{}

sXtrExpandedFmt  Helper command that (protected) fully expands second argument and then applies it to the
                  first, which must be a command that takes a single argument.
2535 \newrobustcmd*{\GlsXtrExpandedFmt}[2]{%
2536   \protected@edef\@glxtr@tmp{#2}%
2537   \expandafter#1\expandafter{\@glxtr@tmp}%
2538 }

tion@counter@or  If in a numbered equation, change the counter to equation. This can be overridden by explic-
                  itly setting the counter in the optional argument of commands like \gls and \glslink.
2539 \newcommand*{\@glxtr@use@equation@counter}{%
2540   \@glxtr@ifnum@mmode{\def\@gls@counter{equation}}{}}%
2541 }

sXtr@do@autoadd  If \GlsXtrAutoAddOnFormat is used, this will automatically use \glsadd. It’s therefore only
                  used with \@gls@link not with \glsadd otherwise it could trigger an infinite loop. The ar-
                  gument indicates the key family (glslink or glossadd).
2542 \newcommand*{\glxtr@do@autoadd}[1]{%

```

AutoAddOnFormat

```
\GlsXtrAutoAddOnFormat[\label][\format list][\glsadd options]
```

If an entry is indexed with the format set to one identified in the comma-separated list, then automatically index it using `\glsadd` with the given options, which may override the current options. Scoping is needed to prevent leakage.

```
2543 \newcommand*{\GlsXtrAutoAddOnFormat}[3][\glslabel]{%
2544   \renewcommand*{\glsxtr@do@autoadd}[1]{%
2545     \begingroup
2546       \protected@edef\@glsxtr@do@autoadd{%
2547         \noexpand\ifstrequal{##1}{\glslink}%
2548         {%
2549           \noexpand\DTLifinlist{\@glsnumberformat}{#2}{\noexpand\glsadd[format={\@glsnumberfor
2550           }%
2551           }%
2552         }%
2553       \@glsxtr@do@autoadd
2554     \endgroup
2555   }%
2556 }
```

`\glslinkwrcontent` This may resolve [issue #189](#) but it may have unexpected consequences. This is currently provided as a trial. If it causes a problem then redefine without the grouping. If no issues are reported it will be added to the base package.

```
2557 \providecommand*{\glslinkwrcontent}[1]{#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`.

```
2558 \def\@gls@link[#1]#2#3{%
2559   \leavevmode

2560   \protected@edef\glslabel{\glsdetoklabel{#2}}%
2561   \def\@gls@link@opts{#1}%
2562   \let\@gls@link@label\glslabel
2563   \let\@glsnumberformat\@glsxtr@defaultnumberformat
2564   \protected@edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
2565   \protected@edef\gls@type{\csname glo@\glslabel @type\endcsname}%
2566   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
```

Save local setting.

```
2567 \@gls@save@glslocal
```

Save current value of `\gls@linkprefix`:

```
2568 \let\@glsxtr@org@gls@linkprefix\gls@linkprefix
```

Initialise `\@glsxtr@local@textformat`

```
2569 \let\@glsxtr@local@textformat\relax
```

Initialise thevalue and theHvalue (v1.19).

```

2570 \def\@glxstr@thevalue{}%
2571 \def\@glxstr@theHvalue{\@glxstr@thevalue}%

```

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

```

2572 \glxstr@trinitwrgloss

```

Initialise whether \hyperlink should be outside \glstextformat (new to v1.21).

```

2573 \glxstr@trinithyperoutside

```

Note that the default link options may override \glxstr@trinitwrgloss.

```

2574 \@gls@setdefault@glslink@opts

```

Increment link counter if enabled (new to v1.26).

```

2575 \glxstr@inc@linkcount

```

Check if the equations option has been set (new to v1.37).

```

2576 \if@glxstr@equations
2577 \glxstr@use@equation@counter
2578 \fi

```

As the original definition.

```

2579 \do@gl@disablehyperinlist
2580 \do@gl@link@checkfirsthyper

```

User hook before options are set (new to v1.26):

```

2581 \glslink@presetkeys

```

Set options.

```

2582 \setkeys{glslink}{#1}%

```

Perform auto add if set (new to v1.37)

```

2583 \glxstr@do@autoadd{glslink}%

```

User hook after options are set:

```

2584 \glslink@postsetkeys

```

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

```

2585 \ifdefempty{\@glxstr@thevalue}%
2586 {%
2587 \gl@saveentrycounter
2588 }%
2589 {%
2590 \let\the@gl@entrycounter\@glxstr@thevalue
2591 \def\the@gl@entrycounter{\@glxstr@theHvalue}%
2592 }%
2593 \gl@setsort{glslabel}%

```

Check if the textformat key has been used.

```

2594 \ifx\@glxstr@local@textformat\relax

```

Check textformat attribute (new to v1.21).

```

2595 \gl@hasattribute{glslabel}{textformat}%
2596 {%

```

```

2597     \protected@edef\@glxstr@attrval{\glsetattribute{\glslabel}{textformat}}}%
2598     \ifcsdef{\@glxstr@attrval}%
2599     {%
2600         \letcs{\@glxstr@textformat}{\@glxstr@attrval}%
2601     }%
2602     {%
2603         \GlossariesExtraWarning{Unknown control sequence name
2604         '@@glxstr@attrval' supplied in textformat attribute
2605         for entry '\glslabel'. Reverting to default \string\glstextformat}%
2606         \let\@glxstr@textformat\glstextformat
2607     }%
2608 }%
2609 {%
2610     \let\@glxstr@textformat\glstextformat
2611 }%
2612 \else
2613     \let\@glxstr@textformat\@glxstr@local@textformat
2614 \fi

```

Encapsulate link text and indexing.

```

2615 \glslinkwrcontent
2616 {%

```

Do write if it should occur before the link text:

```

2617 \ifglxstrinitwrglossbefore
2618     \@do@wrglossary{#2}%
2619 \fi

```

Do the link text:

```

2620 \ifKV@glslink@hyper
2621     \ifglxstr@hyperoutside
2622         \@glslink{\glolinkprefix\glslabel}{\@glxstr@textformat{#3}}%
2623     \else
2624         \@glxstr@textformat{\@glslink{\glolinkprefix\glslabel}{#3}}%
2625     \fi
2626 \else
2627     \ifglxstr@hyperoutside
2628         \glndonohyperlink{\glolinkprefix\glslabel}{\@glxstr@textformat{#3}}%
2629     \else
2630         \@glxstr@textformat{\glndonohyperlink{\glolinkprefix\glslabel}{#3}}%
2631     \fi
2632 \fi

```

Do write if it should occur after the link text:

```

2633 \ifglxstrinitwrglossbefore
2634 \else
2635     \@do@wrglossary{#2}%
2636 \fi
2637 }%

```

Restore original value of \glolinkprefix:

```

2638 \let\glolinkprefix\@glxstr@org@glolinkprefix

```

As the original definition:

```
2639 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
2640 \@gls@restore@glslocal
2641 }
```

```
2642 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}
```

```
2643 \define@key{glossadd}{theHvalue}{\def\@glxtr@theHvalue{#1}}
```

lsaddpresetkeys

```
2644 \newcommand*{\glsaddpresetkeys}{}
```

saddpostsetkeys

```
2645 \newcommand*{\glsaddpostsetkeys}{}
```

\glsadd Redefine to include \@glxtr@record and suppress in headings

```
2646 \renewrobustcmd*{\glsadd}[2][\%
2647   \glxtrifinmark
2648   {}%
2649   {%
2650     \@gls@adjustmode
2651     \begingroup
2652     \@glxtr@record{#1}{#2}{glossadd}%
2653     \glsdoifexists{#2}%
2654     {%
2655       \let\@glsnumberformat\@glxtr@defaultnumberformat
2656       \protected@edef\@gls@counter{\csname glo@glsdetoklabel{#2}@counter\endcsname}%
2657       \def\@glxtr@thevalue{}%
2658       \def\@glxtr@theHvalue{\@glxtr@thevalue}%

```

Implement any default settings (before options are set)

```
2659     \glsaddpresetkeys
2660     \setkeys{glossadd}{#1}%

```

Implement any default settings (after options are set)

```
2661     \glsaddpostsetkeys
2662     \ifdefempty{\@glxtr@thevalue}%
2663     {%
2664       \@gls@saveentrycounter
2665     }%
2666     {%
2667       \let\theglentrycounter\@glxtr@thevalue
2668       \def\theHglentrycounter{\@glxtr@theHvalue}%
2669     }%

```

Define sort key if necessary (in case of sort=use):

```
2670     \@gls@setsort{#2}%

```

Ensure that indexing occurs (since that's the point of `\glsadd`). If indexing has been switched off by default, don't want the setting to affect `\glsadd`. The ignored format `\glsignore` can be used for selection without location, but the indexing still needs to be performed.

```
2671      \KV@glslink@noindexfalse
2672      \@@do@wrglossary{#2}%
2673    }%
2674  \endgroup
2675 }%
2676 }
```

`\glsaddeach` Performs `\glsadd` for each entry listed in the mandatory argument.

```
2677 \newrobustcmd{\glsaddeach}[2][]{%
2678   \@for\@gls@thislabel:=#2\do{\glsadd[#1]{\@gls@thislabel}}%
2679 }
```

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

```
2680 \newcommand*{\@glsxtr@field@linkdefs}{%
2681   \let\glsxtrifwasfirstuse\@secondoftwo
2682   \let\glsifplural\@secondoftwo
2683   \let\glsapscase\@firstofthree
2684   \let\glsinsert\@empty
2685 }
```

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

`assignfieldfont`

```
2686 \newcommand*{\glsxtrassignfieldfont}[1]{%
2687   \ifglseentryexists{#1}%
2688   {%
2689     \ifglshasshort{#1}%
2690     {%
2691       \glssetabbrvfmt{\glscategory{#1}}%
2692       \glsifregular{#1}%
2693       {\let\@gls@field@font\glsxtrregularfont}%
2694       {\let\@gls@field@font\@firstofone}%
2695     }%
2696     {%
2697       \glsifnotregular{#1}%
2698       {\let\@gls@field@font\@firstofone}%
2699       {\let\@gls@field@font\glsxtrregularfont}%
2700     }%
2701   }%
2702   {%
2703     \let\@gls@field@font\@gobble
2704   }%
2705 }
```

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

```
2706 \def\@glstext@#1#2[#3]{%
2707   \glstrassignfieldfont{#2}%
2708   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
2709 }
```

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

```
2710 \def\@GLStext@#1#2[#3]{%
2711   \glstrassignfieldfont{#2}%
2712   \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
2713   {\@gls@field@font{\GLSaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
2714 }
```

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

```
2715 \def\@Glstext@#1#2[#3]{%
2716   \glstrassignfieldfont{#2}%
2717   \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
2718   {\@gls@field@font{\GLSaccesstext{#2}#3}}%
2719 }
```

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

`ecknohyperfirst`

```
2720 \newcommand*{\glstrchecknohyperfirst}[1]{%
2721   \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
2722 }
```

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

```
2723 \def\@glsfirst@#1#2[#3]{%
2724   \glstrassignfieldfont{#2}%
2725   \@gls@field@link
2726   [\let\glstrifwasfirstuse\@firstoftwo
2727   \glstrchecknohyperfirst{#2}%
2728   ]{#1}{#2}%
2729   {\@gls@field@font{\glsaccessfirst{#2}#3}}%
2730 }
```

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

```
2731 \def\@Glsfirst@#1#2[#3]{%
2732   \glstrassignfieldfont{#2}%
2733   \@gls@field@link
2734   [\let\glstrifwasfirstuse\@firstoftwo
2735   \let\glscapscase\@secondofthree
2736   \glstrchecknohyperfirst{#2}%
2737   ]{#1}{#2}%
2738   {\@gls@field@font{\GLSaccessfirst{#2}#3}}%
2739 }
```



```

2737 ]%
2738 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
2739 }

```

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

```

2740 \def\@GLSfirst@#1#2[#3]{%
2741   \glstrassignfieldfont{#2}%
      Ensure that \GLSfirst honours the nohyperfirst attribute.
2742   \@gls@field@link
2743   [\let\glstrifwasfirstuse\@firstoftwo
2744     \let\glscapscase\@thirdofthree
2745     \glstrchecknohyperfirst{#2}%
2746   ]%
2747   {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
2748 }

```

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

```

2749 \def\@glsplural@#1#2[#3]{%
2750   \glstrassignfieldfont{#2}%
2751   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
2752   {\@gls@field@font{\glsaccessplural{#2}#3}}%
2753 }

```

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

```

2754 \def\@Glsplural@#1#2[#3]{%
2755   \glstrassignfieldfont{#2}%
2756   \@gls@field@link
2757   [\let\glsifplural\@firstoftwo
2758     \let\glscapscase\@secondofthree
2759   ]%
2760   {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
2761 }

```

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

```

2762 \def\@GLSplural@#1#2[#3]{%
2763   \glstrassignfieldfont{#2}%
2764   \@gls@field@link
2765   [\let\glsifplural\@firstoftwo
2766     \let\glscapscase\@thirdofthree
2767   ]%
2768   {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
2769 }

```

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

```

2770 \def\glsfirstplural@#1#2[#3]{%
2771   \glstrassignfieldfont{#2}%

```

Ensure that \glsfirstplural honours the nohyperfirst attribute.

```
2772 \@gls@field@link
2773 [\let\glstrifwasfirstuse\@firstoftwo
2774 \let\glsifplural\@firstoftwo
2775 \glstrchecknohyperfirst{#2}%
2776 ]%
2777 {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
2778 }
```

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

```
2779 \def\@Glsfirstplural@#1#2[#3]{%
2780 \glstrassignfieldfont{#2}%
2781 \@gls@field@link
2782 [\let\glstrifwasfirstuse\@firstoftwo
2783 \let\glsifplural\@firstoftwo
2784 \let\glscapscase\@secondofthree
2785 \glstrchecknohyperfirst{#2}%
2786 ]%
2787 {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
2788 }
```

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

```
2789 \def\@GLSfirstplural@#1#2[#3]{%
2790 \glstrassignfieldfont{#2}%
2791 \@gls@field@link
2792 [\let\glstrifwasfirstuse\@firstoftwo
2793 \let\glsifplural\@firstoftwo
2794 \let\glscapscase\@thirdofthree
2795 \glstrchecknohyperfirst{#2}%
2796 ]%
2797 {#1}{#2}%
2798 {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
2799 }
```

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

```
2800 \def\@glsname@#1#2[#3]{%
2801 \glstrassignfieldfont{#2}%
2802 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
2803 }
```

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

```
2804 \def\@Glsname@#1#2[#3]{%
2805 \glstrassignfieldfont{#2}%
2806 \@gls@field@link
2807 [\let\glscapscase\@secondoftwo]{#1}{#2}%
```

```

2808 {\@gls@field@font{\Glsaccessname{#2}#3}}%
2809 }

```

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

```

2810 \def\@GLSname@#1#2[#3]{%
2811   \glstrassignfieldfont{#2}%
2812   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2813   {#1}{#2}%
2814   {\@gls@field@font{\Glsaccessname{#2}\mfirstucMakeUppercase{#3}}}%
2815 }

```

`\@glsdesc@`

```

2816 \def\@glsdesc@#1#2[#3]{%
2817   \glstrassignfieldfont{#2}%
2818   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
2819 }

```

`\@Glsdesc@` First letter uppercase version.

```

2820 \def\@Glsdesc@#1#2[#3]{%
2821   \glstrassignfieldfont{#2}%
2822   \@gls@field@link
2823   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2824   {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
2825 }

```

`\@GLSdesc@` All uppercase version.

```

2826 \def\@GLSdesc@#1#2[#3]{%
2827   \glstrassignfieldfont{#2}%
2828   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2829   {#1}{#2}{\@gls@field@font{\Glsaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2830 }

```

`@glsdescplural@` No case-changing version.

```

2831 \def\@glsdescplural@#1#2[#3]{%
2832   \glstrassignfieldfont{#2}%
2833   \@gls@field@link
2834   [\let\glscapscase\@secondoftwo
2835    \let\glsifplural\@firstoftwo
2836   ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
2837 }

```

`@Glsdescplural@` First letter uppercase version.

```

2838 \def\@Glsdescplural@#1#2[#3]{%
2839   \glstrassignfieldfont{#2}%
2840   \@gls@field@link
2841   [\let\glscapscase\@secondoftwo
2842    \let\glsifplural\@firstoftwo
2843   ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
2844 }

```

@GLSdescplural@ All uppercase version.

```
2845 \def\@GLSdesc@#1#2[#3]{%
2846   \glstrassignfieldfont{#2}%
2847   \@gls@field@link
2848   [\let\glscapscase\@thirdoftwo
2849   \let\glsifplural\@firstoftwo
2850   ]%
2851   {#1}{#2}%
2852   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
2853 }
```

\@glssymbol@

```
2854 \def\@glssymbol@#1#2[#3]{%
2855   \glstrassignfieldfont{#2}%
2856   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
2857 }
```

\@Glsymbol@ First letter uppercase version.

```
2858 \def\@Glsymbol@#1#2[#3]{%
2859   \glstrassignfieldfont{#2}%
2860   \@gls@field@link
2861   [\let\glscapscase\@secondoftwo]%
2862   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
2863 }
```

\@GLSsymbol@ All uppercase version.

```
2864 \def\@GLSsymbol@#1#2[#3]{%
2865   \glstrassignfieldfont{#2}%
2866   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2867   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
2868 }
```

lssymbolplural@ No case-changing version.

```
2869 \def\@lssymbolplural@#1#2[#3]{%
2870   \glstrassignfieldfont{#2}%
2871   \@gls@field@link
2872   [\let\glscapscase\@secondoftwo
2873   \let\glsifplural\@firstoftwo
2874   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2875 }
```

lssymbolplural@ First letter uppercase version.

```
2876 \def\@lssymbolplural@#1#2[#3]{%
2877   \glstrassignfieldfont{#2}%
2878   \@gls@field@link
2879   [\let\glscapscase\@secondoftwo
2880   \let\glsifplural\@firstoftwo
2881   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2882 }
```

LSsymbolplural@ All uppercase version.

```
2883 \def\@GLSsymbol@#1#2[#3]{%
2884   \glstrassignfieldfont{#2}%
2885   \@gls@field@link
2886   [\let\glscapscase\@thirdoftwo
2887   \let\glsifplural\@firstoftwo
2888   ]%
2889   {#1}{#2}%
2890   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2891 }
```

\@Glsuseri@ First letter uppercase version.

```
2892 \def\@Glsuseri@#1#2[#3]{%
2893   \glstrassignfieldfont{#2}%
2894   \@gls@field@link
2895   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2896   {\@gls@field@font{\Glsentryuseri{#2}#3}}%
2897 }
```

\@GLSuseri@ All uppercase version.

```
2898 \def\@GLSuseri@#1#2[#3]{%
2899   \glstrassignfieldfont{#2}%
2900   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2901   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}}%
2902 }
```

\@Glsuserii@ First letter uppercase version.

```
2903 \def\@Glsuserii@#1#2[#3]{%
2904   \glstrassignfieldfont{#2}%
2905   \@gls@field@link
2906   [\let\glscapscase\@secondoftwo]%
2907   {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
2908 }
```

\@GLSuserii@ All uppercase version.

```
2909 \def\@GLSuserii@#1#2[#3]{%
2910   \glstrassignfieldfont{#2}%
2911   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2912   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}}%
2913 }
```

\@Glsuseriii@ First letter uppercase version.

```
2914 \def\@Glsuseriii@#1#2[#3]{%
2915   \glstrassignfieldfont{#2}%
2916   \@gls@field@link
2917   [\let\glscapscase\@secondoftwo]%
2918   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
2919 }
```

```

\@GLSuseriii@ All uppercase version.
2920 \def\@GLSuseriii@#1#2[#3]{%
2921   \glstrassignfieldfont{#2}%
2922   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2923   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
2924 }

\@GLSuseriv@ First letter uppercase version.
2925 \def\@GLSuseriv@#1#2[#3]{%
2926   \glstrassignfieldfont{#2}%
2927   \@gls@field@link
2928   [\let\glscapscase\@secondoftwo]%
2929   {#1}{#2}{\@gls@field@font{\Glentryuseriv{#2}#3}}}%
2930 }

\@GLSuseriv@ All uppercase version.
2931 \def\@GLSuseriv@#1#2[#3]{%
2932   \glstrassignfieldfont{#2}%
2933   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2934   {#1}{#2}%
2935   {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
2936 }

\@GLSuserv@ First letter uppercase version.
2937 \def\@GLSuserv@#1#2[#3]{%
2938   \glstrassignfieldfont{#2}%
2939   \@gls@field@link
2940   [\let\glscapscase\@secondoftwo]%
2941   {#1}{#2}{\@gls@field@font{\Glentryuserv{#2}#3}}}%
2942 }

\@GLSuserv@ All uppercase version.
2943 \def\@GLSuserv@#1#2[#3]{%
2944   \glstrassignfieldfont{#2}%
2945   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2946   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
2947 }

\@GLSuservi@ First letter uppercase version.
2948 \def\@GLSuservi@#1#2[#3]{%
2949   \glstrassignfieldfont{#2}%
2950   \@gls@field@link
2951   [\let\glscapscase\@secondoftwo]%
2952   {#1}{#2}{\@gls@field@font{\Glentryuservi{#2}#3}}}%
2953 }

\@GLSuservi@ All uppercase version.
2954 \def\@GLSuservi@#1#2[#3]{%

```

```

2955 \glstrassignfieldfont{#2}%
2956 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2957 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glstentryuservi{#2}{#3}}}%
2958 }

```

Commands like `\acrshort` already set `\glstifplural`, but they don't set `\glstxtrifwasfirstuse` so they need adjusting. These commands shouldn't be used with `\newabbreviation`, but the redefinitions below allow for users reverting `\newacronym` back to its base definition.

`\base@acrcmd@warn` Warn user that they need to use to new abbreviation commands.

```

2959 \newcommand*{\@@glstr@base@acrcmd@warn}[2]{%
2960 \GlossariesExtraWarning{Base acronym command \string#1\space
2961 should not be used with new abbreviation definitions. Use
2962 \string#2\space instead}%
2963 }

```

`\xtr@base@acrcmd` Warn user that they need to use to new abbreviation commands.

```

2964 \let\@glstr@base@acrcmd\@@glstr@base@acrcmd@warn

```

`\@acrshort` No case change.

```

2965 \def\@acrshort#1#2[#3]{%
2966 \@glstr@base@acrcmd\acrshort\glstrshort
2967 \glstoifexists{#2}%
2968 {%
2969 \let\do@glst@link@checkfirsthyper\@glst@link@nocheckfirsthyper
2970 \let\glstxtrifwasfirstuse\@secondoftwo
2971 \let\glstifplural\@secondoftwo
2972 \let\glscapscase\@firstofthree
2973 \let\glstinsert\@empty
2974 \def\glscustomtext{%
2975 \acronymfont{\glstaccessshort{#2}}#3%
2976 }%
2977 \@glst@link[#1]{#2}{\csname glst@glstype @entryfmt\endcsname}%
2978 }%
2979 \glspostlinkhook
2980 }

```

`\@Acrshort` First letter uppercase.

```

2981 \def\@Acrshort#1#2[#3]{%
2982 \@glstr@base@acrcmd\Acrshort\Glsxtrshort
2983 \glstoifexists{#2}%
2984 {%
2985 \let\do@glst@link@checkfirsthyper\@glst@link@nocheckfirsthyper
2986 \let\glstxtrifwasfirstuse\@secondoftwo
2987 \let\glstifplural\@secondoftwo
2988 \let\glscapscase\@secondofthree
2989 \let\glstinsert\@empty
2990 \def\glscustomtext{%
2991 \acronymfont{\Glstaccessshort{#2}}#3%

```

```

2992     }%
2993     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2994 }%
2995 \glspostlinkhook
2996 }

```

\@ACRshort All uppercase.

```

2997 \def\@ACRshort#1#2[#3]{%
2998   \@glsxtr@base@acrcmd\ACRshort\GLSxtrshort
2999   \glsdoifexists{#2}%
3000   {%
3001     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3002     \let\glsxtrifwasfirstuse\@secondoftwo
3003     \let\glsifplural\@secondoftwo
3004     \let\glscapscase\@thirdofthree
3005     \let\glsinsert\@empty
3006     \def\glscustomtext{%
3007       \mfirstucMakeUppercase{\acronymfont{\glsaccessshort{#2}}#3}%
3008     }%
3009     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3010   }%
3011   \glspostlinkhook
3012 }

```

\@acrshortpl No case change.

```

3013 \def\@acrshortpl#1#2[#3]{%
3014   \@glsxtr@base@acrcmd\acrshortpl\glsxtrshortpl
3015   \glsdoifexists{#2}%
3016   {%
3017     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3018     \let\glsxtrifwasfirstuse\@secondoftwo
3019     \let\glsifplural\@firstoftwo
3020     \let\glsapspace\@firstofthree
3021     \let\glsinsert\@empty
3022     \def\glscustomtext{%
3023       \acronymfont{\glsaccessshortpl{#2}}#3%
3024     }%
3025     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3026   }%
3027   \glspostlinkhook
3028 }

```

\@Acrshortpl First letter uppercase.

```

3029 \def\@Acrshortpl#1#2[#3]{%
3030   \@glsxtr@base@acrcmd\Acrshortpl\Glsxtrshortpl
3031   \glsdoifexists{#2}%
3032   {%
3033     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3034     \let\glsxtrifwasfirstuse\@secondoftwo

```



```

3035 \let\glsifplural\@firstoftwo
3036 \let\glscapscase\@secondofthree
3037 \let\glsinsert\@empty
3038 \def\glscustomtext{%
3039 \acronymfont{\Glsaccessshortpl{#2}}#3%
3040 }%
3041 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3042 }%
3043 \glspostlinkhook
3044 }

```

\@ACRshortpl All uppercase.

```

3045 \def\@ACRshortpl#1#2[#3]{%
3046 \@glsxtr@base@acrcmd\ACRshortpl\GLSxtrshortpl
3047 \glsdoifexists{#2}%
3048 {%
3049 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3050 \let\glsxtrifwasfirstuse\@secondoftwo
3051 \let\glsifplural\@firstoftwo
3052 \let\glsapscase\@thirdofthree
3053 \let\glsinsert\@empty
3054 \def\glscustomtext{%
3055 \mfirstucMakeUppercase{\acronymfont{\Glsaccessshortpl{#2}}#3}%
3056 }%
3057 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3058 }%
3059 \glspostlinkhook
3060 }

```

\@acrlong No case change.

```

3061 \def\@acrlong#1#2[#3]{%
3062 \@glsxtr@base@acrcmd\acrlong\glsxtrlong
3063 \glsdoifexists{#2}%
3064 {%
3065 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3066 \let\glsxtrifwasfirstuse\@secondoftwo
3067 \let\glsifplural\@secondoftwo
3068 \let\glsapscase\@firstofthree
3069 \let\glsinsert\@empty
3070 \def\glscustomtext{%
3071 \acronymfont{\Glsaccesslong{#2}}#3%
3072 }%
3073 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3074 }%
3075 \glspostlinkhook
3076 }

```

\@Acrlong First letter uppercase.

```

3077 \def\@Acrlong#1#2[#3]{%

```

```

3078 \@glsxtr@base@acrcmd\Acrlong\Glsxtrlong
3079 \glsdoifexists{#2}%
3080 {%
3081   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3082   \let\glsxtrifwasfirstuse\@secondoftwo
3083   \let\glsifplural\@secondoftwo
3084   \let\gls caps case\@secondofthree
3085   \let\glsinsert\@empty
3086   \def\gls custom text{%
3087     \acronymfont{\Gls access long{#2}}#3%
3088   }%
3089   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3090 }%
3091 \gls post link hook
3092 }

```

\@ACRlong All uppercase.

```

3093 \def\@ACRlong#1#2[#3]{%
3094   \@glsxtr@base@acrcmd\ACRlong\GLSxtrlong
3095   \glsdoifexists{#2}%
3096   {%
3097     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3098     \let\glsxtrifwasfirstuse\@secondoftwo
3099     \let\glsifplural\@secondoftwo
3100     \let\gls caps case\@thirdofthree
3101     \let\glsinsert\@empty
3102     \def\gls custom text{%
3103       \mfirstucMakeUppercase{\acronymfont{\Gls access long{#2}}#3}%
3104     }%
3105     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3106   }%
3107   \gls post link hook
3108 }

```

\@acrlongpl No case change.

```

3109 \def\@acrlongpl#1#2[#3]{%
3110   \@glsxtr@base@acrcmd\acrlongpl\glsxtrlongpl
3111   \glsdoifexists{#2}%
3112   {%
3113     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3114     \let\glsxtrifwasfirstuse\@secondoftwo
3115     \let\glsifplural\@firstoftwo
3116     \let\gls caps case\@firstofthree
3117     \let\glsinsert\@empty
3118     \def\gls custom text{%
3119       \acronymfont{\Gls access longpl{#2}}#3%
3120     }%
3121     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3122   }%

```

```

3123 \glspostlinkhook
3124 }

```

\@Acrlongpl First letter uppercase.

```

3125 \def\@Acrlongpl#1#2[#3]{%
3126 \@glstr@base@acrcmd\Acrlongpl\Glsxtrlongpl
3127 \glsoifexists{#2}%
3128 {%
3129 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3130 \let\glstrifwasfirstuse\@secondoftwo
3131 \let\gl@ifplural\@firstoftwo
3132 \let\glscapscase\@secondofthree
3133 \let\glinsert\@empty
3134 \def\glscustomtext{%
3135 \acronymfont{\Glsaccesslongpl{#2}}#3%
3136 }%
3137 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3138 }%
3139 \glspostlinkhook
3140 }

```

\@ACRlongpl All uppercase.

```

3141 \def\@ACRlongpl#1#2[#3]{%
3142 \@glstr@base@acrcmd\ACRlongpl\GLSxtrlongpl
3143 \glsoifexists{#2}%
3144 {%
3145 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
3146 \let\glstrifwasfirstuse\@secondoftwo
3147 \let\gl@ifplural\@firstoftwo
3148 \let\glscapscase\@thirdofthree
3149 \let\glinsert\@empty
3150 \def\glscustomtext{%
3151 \mfirstucMakeUppercase{\acronymfont{\Glsaccesslongpl{#2}}#3}%
3152 }%
3153 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3154 }%
3155 \glspostlinkhook
3156 }

```

The full formats use the internal long and short commands (such as \@acrshort and \@acrlong). Therefore they don't need adjustments, but they do need clearer warnings. This means three warnings per use (once for the full command and once each for the short and long commands), but at least this way the most important warning (replace \acrfull with \glstrfull etc) is present.

\@acrfull

```

3157 \def\@acrfull#1#2[#3]{%
3158 \@glstr@base@acrcmd\acrfull\glstrfull

```

```

3159 \acrfullfmt{#1}{#2}{#3}%
3160 }

```

\@Acrfull

```

3161 \def\@Acrfull#1#2[#3]{%
3162   \@glsxtr@base@acrcmd\Acrfull\Glsxtrfull
3163   \acrfullfmt{#1}{#2}{#3}%
3164 }

```

\@ACRfull

```

3165 \def\@ACRfull#1#2[#3]{%
3166   \@glsxtr@base@acrcmd\ACRfull\GLSxtrfull
3167   \ACRfullfmt{#1}{#2}{#3}%
3168 }

```

\@acrfullpl

```

3169 \def\@acrfullpl#1#2[#3]{%
3170   \@glsxtr@base@acrcmd\acrfullpl\glsxtrfullpl
3171   \acrfullplfmt{#1}{#2}{#3}%
3172 }

```

\@Acrfullpl

```

3173 \def\@Acrfullpl#1#2[#3]{%
3174   \@glsxtr@base@acrcmd\Acrfullpl\Glsxtrfullpl
3175   \Acrfullplfmt{#1}{#2}{#3}%
3176 }

```

\@ACRfullpl

```

3177 \def\@ACRfullpl#1#2[#3]{%
3178   \@glsxtr@base@acrcmd\ACRfullpl\GLSxtrfullpl
3179   \ACRfullplfmt{#1}{#2}{#3}%
3180 }

```

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

\@glsaddkey

```

3181 \renewcommand*{\@glsaddkey}[7]{%
3182   \key@ifundefined{glossentry}{#1}%
3183   {%
3184     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
3185     \appto\@gls@keymap{,{#1}{#1}}%
3186     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
3187     \appto\@newglossaryentryposthook{%
3188       \letcs{\@glo@tmp}{@glo@#1}%
3189       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
3190     }%
3191     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
3192     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

3193 \ifcsdef{@gls@user@#1@}%
3194 {%
3195   \PackageError{glossaries}%
3196   {Can't define '\string#5' as helper command
3197   '\expandafter\string\csname @gls@user@#1@\endcsname' already
3198   exists}%
3199   }%
3200 }%
3201 {%
3202   \expandafter\newcommand\expandafter*\expandafter
3203   {\csname @gls@user@#1\endcsname}[2][ ]{%
3204     \new@ifnextchar[%
3205       {\csuse{@gls@user@#1@}{##1}{##2}}%
3206       {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
3207   \csdef{@gls@user@#1@}##1##2[##3]{%
3208     \@gls@field@link{##1}{##2}{#3{##2}##3}%
3209   }%
3210   \newrobustcmd*{#5}{%
3211     \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
3212   }%

```

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

```

3213 \ifcsdef{@Gls@user@#1@}%
3214 {%
3215   \PackageError{glossaries}%
3216   {Can't define '\string#6' as helper command
3217   '\expandafter\string\csname @Gls@user@#1@\endcsname' already
3218   exists}%
3219   }%
3220 }%
3221 {%
3222   \expandafter\newcommand\expandafter*\expandafter
3223   {\csname @Gls@user@#1\endcsname}[2][ ]{%
3224     \new@ifnextchar[%
3225       {\csuse{@Gls@user@#1@}{##1}{##2}}%
3226       {\csuse{@Gls@user@#1@}{##1}{##2}[ ]}}%
3227   \csdef{@Gls@user@#1@}##1##2[##3]{%
3228     \@gls@field@link[\let\glscaps@case\@secondofthree]%
3229     {##1}{##2}{#4{##2}##3}%
3230   }%
3231   \newrobustcmd*{#6}{%
3232     \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
3233   }%

```

Finally the all caps version (modified):

```

3234 \ifcsdef{@GLS@user@#1@}%
3235 {%
3236   \PackageError{glossaries}%
3237   {Can't define '\string#7' as helper command

```

```

3238      '\expandafter\string\csname @GLS@user@#1@\endcsname' already
3239      exists}%
3240    {%
3241  }%
3242  {%
3243    \expandafter\newcommand\expandafter*\expandafter
3244    {\csname @GLS@user@#1@\endcsname}[2][\%
3245      \new@ifnextchar[%
3246        {\csuse{@GLS@user@#1@}{##1}{##2}}%
3247        {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%
3248    \csdef{@GLS@user@#1@}##1##2[##3]{%
3249      \gls@field@link[\let\glscapscase\@thirdofthree]%
3250      {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
3251    }%
3252    \newrobustcmd*{#7}{%
3253      \expandafter\@gls@hyp@opt\csname @GLS@user@#1@\endcsname}%
3254    }%
3255  }%
3256  {%
3257    \PackageError{glossaries-extra}{Key '#1' already exists}{}%
3258  }%
3259 }

```

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

```

3260 \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.

```

3261 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
3262 \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 to automatically set the value for the commands that change the first use flag. The other commands should set \glsxtrifwasfirstuse to \@secondoftwo (which is done in \@glsxtr@field@linkdefs). Note that if the entry is undefined (as with bib2gls on the first \LaTeX run), \ifglsused does neither true nor false parts. However, in that case, this macro won't be called anyway (since it's used in the argument of \glsdoifexistsordo).

```

3263 \ifglsused{\glslabel}%
3264 {\let\glsxtrifwasfirstuse\@secondoftwo}
3265 {\let\glsxtrifwasfirstuse\@firstoftwo}%

```

Store the category label for convenience.

```

3266 \protected@edef\glscategorylabel{\glscategory{\glslabel}}%
3267 \ifglsused{\glslabel}%
3268 {%
3269   \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
3270   {\KV@glslink@hyperfalse}}}%

```

```

3271 }%
3272 {%
3273   \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
3274   {\KV@glslink@hyperfalse}}}%
3275 }%
3276 \glslinkcheckfirsthyperhook
3277 }

```

`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.

```

3278 \ifdef\do@glsglsdisablehyperinlist
3279 {%
3280   \let\@glsextr@do@glsglsdisablehyperinlist\do@glsglsdisablehyperinlist
3281   \renewcommand*{\do@glsglsdisablehyperinlist}{%
3282     \@glsextr@do@glsglsdisablehyperinlist
3283     \glsifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}}}%
3284   }
3285 }
3286 {}

```

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

```

3287 \define@boolkey{glslink}{noindex}[true]{}
3288 \KV@glslink@noindexfalse

```

`s@save@glsllocal` Defined in glossaries v4.50 so may not be defined.

```

3289 \providecommand*{\@glsgls@save@glsllocal}{%
3290   \let\if@org@KV@glslink@local\ifKV@glslink@local
3291 }

```

`estore@glsllocal` Defined in glossaries v4.50 so may not be defined.

```

3292 \providecommand*{\@glsgls@restore@glsllocal}{%
3293   \ifKV@glslink@local
3294     \let\@glsgls@do@glsglsunset\glsllocalunset
3295   \else
3296     \let\@glsgls@do@glsglsunset\glsglsunset
3297   \fi
3298 }

```

`glsgls@do@glsglsunset` Defined in glossaries v4.50 so may not be defined.

```

3299 \providecommand*{\@glsgls@do@glsglsunset}[1]{\glsglsunset{#1}}

```

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

`lt@glslink@opts`

```

3300 \ifdef\@glsgls@setdefault@glslink@opts
3301 {
3302   \renewcommand*{\@glsgls@setdefault@glslink@opts}{%

```

```

3303 \KV@glslink@noindexfalse
3304 \@glxtrsetaliasnoindex
3305 }
3306 }
3307 {
    Not defined so prepend it to \do@glstdisablehyperinlist to achieve the same effect.
3308 \newcommand*{\@gl@setdefault@glslink@opts}{%
3309 \KV@glslink@noindexfalse
3310 \@glxtrsetaliasnoindex
3311 }
3312 \preto\do@glstdisablehyperinlist{\@gl@setdefault@glslink@opts}
3313 }

```

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.)

```

3314 \providecommand*\glxtrsetaliasnoindex{%
3315 \KV@glslink@noindextrue
3316 }

```

setaliasnoindex The change made in v1.46 to remove the grouping has had the knock-on effect of redefining \glscurrentfieldvalue, which may be a problem, so v1.47 has changed this to use \ifcsvoid.

```

3317 \newcommand*{\@glxtrsetaliasnoindex}{%
3318 \ifcsvoid{glo@glstdetoklabel{\glslabel}@alias}%
3319 {}%
3320 {%
3321 \let\glxtrindexaliased\@glxtrindexaliased
3322 \glxtrsetaliasnoindex
3323 \let\glxtrindexaliased\@no@glxtrindexaliased
3324 }%
3325 }

```

xtrindexaliased

```

3326 \newcommand{\@glxtrindexaliased}{%
3327 \ifKV@glslink@noindex
3328 \else
3329 \begingroup
3330 \let\@gl@numberformat\@glxtr@defaultnumberformat
3331 \protected@edef\@gl@counter{\csname glo@glstdetoklabel{\glslabel}@counter\endcsname}%
3332 \glxtr@saveentrycounter
3333 \@@do@wrglossary{\glxtralias{\glslabel}}%
3334 \endgroup
3335 \fi
3336 }

```

xtrindexaliased


```

3337 \newcommand{\@no@glxtrindexaliased}{%
3338   \PackageError{glossaries-extra}{\string\glxtrindexaliased\space
3339   not permitted outside definition of \string\glxtrsetaliasnoindex}%
3340   {}%
3341 }

```

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

```

3342 \let\glxtrindexaliased\@no@glxtrindexaliased

```

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

```

3343 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
3344   \renewcommand*{\@gls@setdefault@glslink@opts}{%
3345     \setkeys{glslink}{#1}%
3346     \@glxtrsetaliasnoindex
3347   }%
3348 }

```

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

```

3349 \newcommand*{\glxtrifindexing}[2]{%
3350   \ifKV@glslink@noindex #2\else #1\fi
3351 }

```

`\glswriteentry` Redefine to test for `indexonlyfirst` category attribute. This needs to use `\GlsXtrIfUnusedOrUndefined` instead of `\ifglused` to allow it to work with `bib2gls`.

```

3352 \renewcommand*{\glswriteentry}[2]{%
3353   \glxtrifindexing
3354   {%
3355     \ifglindexonlyfirst
3356       \GlsXtrIfUnusedOrUndefined{#1}
3357       {#2}%
3358     {\glxtrdoautoindexname{#1}{dualindex}}%
3359     \else
3360       \gl@ifattribute{#1}{indexonlyfirst}{true}%
3361       {%
3362         \GlsXtrIfUnusedOrUndefined{#1}%
3363         {#2}%
3364         {\glxtrdoautoindexname{#1}{dualindex}}%
3365       }%
3366     {#2}%
3367   \fi
3368   }%
3369   {}%
3370 }

```

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

```

3371 \appto\do@wrglossary{\@glxtr@do@wrindex
3372   \glxtrdowrglossaryhook{\@gls@label}%
3373 }

```

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

Similarly for the “noidx” version:

s@noidxglossary

```
3374 \appto\gls@noidxglossary{\@glsxtr@do@@wrindex
3375   \glsxtrdowrglossaryhook{\@gls@label}}%
3376 }
```

xtr@do@@wrindex

```
3377 \newcommand*{\@glsxtr@do@@wrindex}{%
3378   \glsxtrdoautoindexname{\@gls@label}{dualindex}}%
3379 }
```

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.)

```
3380 \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.

```
3381 \newcommand*{\@gls@alt@hyp@opt}[1]{%
3382   \let\glslinkvar\@firstofthree
3383   \let\@gls@hyp@opt@cs#1\relax
3384   \@ifstar{\s@gls@hyp@opt}%
3385   {\@ifnextchar+%
3386     {\@firstoftwo{\p@gls@hyp@opt}}%
3387     {%
3388       \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
3389       {\@firstoftwo{\@alt@gls@hyp@opt}}%
3390       {#1}}%
3391   }%
3392 }%
3393 }
```

alt@gls@hyp@opt User version

```
3394 \newcommand*{\@alt@gls@hyp@opt}[1][ ]{%
3395   \let\glslinkvar\@firstofthree
3396   \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}
```

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

```
3397 \newcommand*{\@gls@alt@hyp@opt@char}{}
```

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

```
3398 \newcommand*{\@gls@alt@hyp@opt@keys}{}
```

rSetAltModifier

```
3399 \newcommand*{\GlsXtrSetAltModifier}[2]{%
3400   \let\@gls@hyp@opt\@gls@alt@hyp@opt
```

Check that the supplied character isn't + or *

```

3401 \ifstrequal{#1}{+}%
3402 {\PackageError{glossaries-extra}%
3403   {Can't use '#1' as modifier (it's already in use)}}{}%
3404 {%
3405   \ifstrequal{#1}{*}%
3406   {\PackageError{glossaries-extra}%
3407     {Can't use '#1' as modifier (it's already in use)}}{}%
3408   }%
3409 }%
3410 \def\@gls@alt@hyp@opt@char{#1}%
3411 \def\@gls@alt@hyp@opt@keys{#2}%
3412 \ifdefequal\@glsxtr@record@setting\@glsxtr@record@setting@off
3413 {}%
3414 {%

```

Let **bib2gls** know the modifier.

```

3415 \protected@write\@auxout{}{\string\providecommand{\string\@glsxtr@altmodifier}[1]{}%
3416 \protected@write\@auxout{}{\string\@glsxtr@altmodifier{#1}}%
3417 }%
3418 }

```

org@dohyperlink

```

3419 \let\glsxtr@org@dohyperlink\glsdohyperlink

```

glsnavhyperlink

Since `\glsnavhyperlink` uses `\@glslink`, it's necessary to patch it uses `\glsdohyperlink` instead of `\glsxtrdohyperlink`. The simplest way to achieve this is to locally let `\glsxtrdohyperlink` to `\glsdohyperlink`.

This command is provided by `glossary-hypernav` so it may not exist.

```

3420 \ifdef\glsnavhyperlink
3421 {
3422   \renewcommand*{\glsnavhyperlink}[3][\@glo@type]{%
3423     \protected@edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%

```

Scope:

```

3424   {%
3425     \let\glsxtrdohyperlink\glsxtr@org@dohyperlink
3426     \@glslink{\glsnavhyperlinkname{#1}{#2}}{#3}%
3427   }%
3428 }%
3429 }
3430 {}

```

Patch if `glossaries` pre 4.50.

```

3431 \ifdef\@@gls@navhypertarget
3432 {}
3433 {%

```

snavhypertarget

```
3434 \renewcommand*{\glsnavhypertarget}{\protect\@gls@snavhypertarget}
```

@navhypertarget

```
3435 \newcommand*{\@gls@snavhypertarget}[3][\@gls@type]{%
```

```
3436 \@glsnavhypertarget{#1}{#2}{#3}%
```

```
3437 }
```

```
3438 }%
```

snavhypertarget Similarly for \@glsnavhypertarget

```
3439 \ifdef\@glsnavhypertarget
```

```
3440 {%
```

```
3441 \renewcommand*{\@glsnavhypertarget}[3]{%
```

```
3442 \protected@write\@auxout{}{\string\@gls@hypergroup{#1}{#2}}%
```

```
3443 \@glsxtr@do@org@target{\glsnavhyperlinkname{#1}{#2}}{#3}%
```

```
3444 \ifcsdef{\@gls@hypergroup@list@#1}%
```

```
3445 {%
```

```
3446 \letcs\@gls@list{\@gls@hypergroup@list@#1}%
```

```
3447 \protected@edef\@gls@thishypernavlabel{#2}%
```

```
3448 \expandafter\DTLifinlist\expandafter{\@gls@thishypernavlabel}\@gls@list{}}%
```

```
3449 {%
```

```
3450 \GlossariesWarningNoLine{Navigation panel
```

```
3451 for glossary type ‘#1’^^Jmissing group ‘#2’}%
```

```
3452 \gdef\@gls@hypergroup@prerun{%
```

```
3453 \GlossariesWarningNoLine{Navigation panel
```

```
3454 has changed. Rerun LaTeX}}%
```

```
3455 }%
```

```
3456 }%
```

```
3457 {%
```

```
3458 \GlossariesWarningNoLine{Navigation panel
```

```
3459 for glossary type ‘#1’^^Jmissing group ‘#2’}%
```

```
3460 \gdef\@gls@hypergroup@prerun{%
```

```
3461 \GlossariesWarningNoLine{Navigation panel
```

```
3462 has changed. Rerun LaTeX}}%
```

```
3463 }%
```

```
3464 }%
```

```
3465 }
```

```
3466 {}
```

The redefinition of `\glsdohyperlink` has been causing problems so introduce a new command instead.

sxtrdohyperlink

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

and `\glxtrshort` to use `\glstryshort` and, similarly, the long form commands like `\acrlong` and `\glxtrlong` to use `\glstrylong`. Added attribute check.

```

3467 \newcommand*{\glxtrdohyperlink}[2]{%
3468   \glshasattribute{\glslabel}{targeturl}%
3469   {%
3470     \glshasattribute{\glslabel}{targetname}%
3471     {%
3472       \glshasattribute{\glslabel}{targetcategory}%
3473       {%
3474         \hyperref{\glsggetattribute{\glslabel}{targeturl}}{%
3475           {\glsggetattribute{\glslabel}{targetcategory}}}%
3476         {\glsggetattribute{\glslabel}{targetname}}}%
3477         {\glxtrprotectlinks#2}}%
3478       }%
3479     {%
3480       \hyperref{\glsggetattribute{\glslabel}{targeturl}}{%
3481         }%
3482         {\glsggetattribute{\glslabel}{targetname}}}%
3483         {\glxtrprotectlinks#2}}%
3484       }%
3485     }%
3486   {%
3487     \href{\glsggetattribute{\glslabel}{targeturl}}{%
3488       {\glxtrprotectlinks#2}}%
3489     }%
3490   }%
3491   {%

```

Check for alias.

```

3492   \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
3493   \ifdefvoid\gloaliaslabel
3494   {%
3495     \glxtrhyperlink{#1}{\glxtrprotectlinks#2}}%
3496   }%
3497   {%

```

Is the alias a multi-entry?

```

3498     \glxtrifmulti\gloaliaslabel
3499     {%

```

Get the main target.

```

3500       \letcs\gloaliaslabel{@gls@combined@\gloaliaslabel @main}%
3501     }%
3502   {}%

```

Redirect link to the alias target.

```

3503     \glxtrhyperlink
3504     {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
3505     {\glxtrprotectlinks#2}}%
3506   }%
3507 }%

```

```

3508 }

glsxtrhyperlink  Allows integration with the base glossaries package's debug=showtargets option.
3509 \newcommand{\glsxtrhyperlink}[2]{%
3510   \glsdoshowtarget{#1}{\hyperlink{#1}{#2}}%
3511 }%

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.
3512 \renewrobustcmd*{\glsxtrhyperlink}[2][\glsentrytext{\@glo@label}]{%
3513   \glsdoifexists{#2}%
3514   {%
3515     \def\@glo@label{#2}%
3516     {\protected@edef\glslabel{#2}%
3517       \@glslink{\glslinkprefix\glslabel}{#1}}%
3518   }%
3519 }

glsdisablehyper  Redefine in case we have an old version of glossaries. This now uses \def rather than \let to
                  allow for redefinitions of \glsdonohyperlink.
3520 \renewcommand{\glsdisablehyper}{%
3521   \KV@glslink@hyperfalse
3522   \def\@glslink{\glsdonohyperlink}%
3523   \let\@glstarget\@secondoftwo
3524 }

\glsenablehyper  This now uses \def rather than \let to allow for redefinitions of \glsdohypertarget and
                  \glsdohyperlink.
3525 \renewcommand{\glsenablehyper}{%
3526   \KV@glslink@hypertrue
3527   \def\@glslink{\glsxtrdohyperlink}%
3528   \def\@glstarget{\glsdohypertarget}%
3529 }

glsdonohyperlink  This command was only introduced in glossaries v4.20, so it may not be defined (therefore
                  use \def). 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 (the link text in
                  \hyperlink is also scoped, so it's consistent).
3530 \def\glsdonohyperlink#1#2{{\glsxtrprotectlinks #2}}

\@glslink  Reset \@glslink with patched versions:
3531 \ifcsundef{hyperlink}%
3532 {%
3533   \def\@glslink{\glsdonohyperlink}
3534 }%

```

```

3535 {%
3536   \def\@glslink{\glsxtrdohyperlink}
3537 }

```

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

```

3538 \newcommand*{\glsxtrprotectlinks}{%
3539   \KV@glslink@hyperfalse
3540   \KV@glslink@noindextrue
3541   \let\@gls@\@glsxtr@p@text@
3542   \let\@Gls@\@Glsxtr@p@text@
3543   \let\@GLS@\@GLSxtr@p@text@
3544   \let\@glspl@\@glsxtr@p@plural@
3545   \let\@Glspl@\@Glsxtr@p@plural@
3546   \let\@GLSpl@\@GLSxtr@p@plural@
3547   \let\@glsxtrshort@\@glsxtr@p@short@
3548   \let\@Glsxtrshort@\@Glsxtr@p@short@
3549   \let\@GLSxtrshort@\@GLSxtr@p@short@
3550   \let\@glsxtrlong@\@glsxtr@p@long@
3551   \let\@Glsxtrlong@\@Glsxtr@p@long@
3552   \let\@GLSxtrlong@\@GLSxtr@p@long@
3553   \let\@glsxtrshortpl@\@glsxtr@p@shortpl@
3554   \let\@Glsxtrshortpl@\@Glsxtr@p@shortpl@
3555   \let\@GLSxtrshortpl@\@GLSxtr@p@shortpl@
3556   \let\@glsxtrlongpl@\@glsxtr@p@longpl@
3557   \let\@Glsxtrlongpl@\@Glsxtr@p@longpl@
3558   \let\@GLSxtrlongpl@\@GLSxtr@p@longpl@
3559   \let\@acrshort@\@glsxtr@p@acrshort@
3560   \let\@Acrshort@\@Glsxtr@p@acrshort@
3561   \let\@ACRshort@\@GLSxtr@p@acrshort@
3562   \let\@acrshortpl@\@glsxtr@p@acrshortpl@
3563   \let\@Acrshortpl@\@Glsxtr@p@acrshortpl@
3564   \let\@ACRshortpl@\@GLSxtr@p@acrshortpl@
3565   \let\@acrlong@\@glsxtr@p@acrlong@
3566   \let\@Acrlong@\@Glsxtr@p@acrlong@
3567   \let\@ACRlong@\@GLSxtr@p@acrlong@
3568   \let\@acrlongpl@\@glsxtr@p@acrlongpl@
3569   \let\@Acrlongpl@\@Glsxtr@p@acrlongpl@
3570   \let\@ACRlongpl@\@GLSxtr@p@acrlongpl@
3571 }

```

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

`@glsxtr@p@text@`

```

3572 \def\@glsxtr@p@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}

```

`@Glsxtr@p@text@`

```

3573 \def\@Glsxtr@p@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}

```

```

@GLSxtr@p@text@
3574 \def\@GLSxtr@p@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}}

lsxtr@p@plural@
3575 \def\@glsxtr@p@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}}

lsxtr@p@plural@
3576 \def\@Glsxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}}

LSxtr@p@plural@
3577 \def\@GLSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}}

glsxtr@p@short@
3578 \def\@glsxtr@p@short@#1#2[#3]{%
3579 {%
3580   \glsssetabbrvfmt{\glscategory{#2}}}%
3581   \glsabbrvfont{\glsentryshort{#2}}#3}%
3582 }%
3583 }

Glsxtr@p@short@
3584 \def\@Glsxtr@p@short@#1#2[#3]{%
3585 {%
3586   \glsssetabbrvfmt{\glscategory{#2}}}%
3587   \glsabbrvfont{\Glsentryshort{#2}}#3}%
3588 }%
3589 }

GLSxtr@p@short@
3590 \def\@GLSxtr@p@short@#1#2[#3]{%
3591 {%
3592   \glsssetabbrvfmt{\glscategory{#2}}}%
3593   \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshort{#2}}#3}%
3594   }%
3595 }

sxtr@p@shortpl@
3596 \def\@glsxtr@p@shortpl@#1#2[#3]{%
3597 {%
3598   \glsssetabbrvfmt{\glscategory{#2}}}%
3599   \glsabbrvfont{\glsentryshortpl{#2}}#3}%
3600 }%
3601 }

sxtr@p@shortpl@
3602 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
3603 {%
3604   \glsssetabbrvfmt{\glscategory{#2}}}%

```



```

3605 \glsabbrvfont{\Glsentryshortpl{#2}}#3%
3606 }%
3607 }

```

Sxtr@p@shortpl@

```

3608 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
3609   {%
3610     \glssetabbrvfmt{\glscategory{#2}}%
3611     \mfirstucMakeUppercase{\glsabbrvfont{\Glsentryshortpl{#2}}#3}%
3612   }%
3613 }

```

@glxtr@p@long@

```

3614 \def\@glxtr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}

```

@Glsxtr@p@long@

```

3615 \def\@Glsxtr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}

```

@GLSxtr@p@long@

```

3616 \def\@GLSxtr@p@long@#1#2[#3]{%
3617   {\mfirstucMakeUppercase{\glsfont{\Glsentrylong{#2}}#3}}

```

lsxtr@p@longpl@

```

3618 \def\@lsxtr@p@longpl@#1#2[#3]{\Glsentrylongpl{#2}#3}

```

lSxtr@p@longpl@

```

3619 \def\@lSxtr@p@longpl@#1#2[#3]{\glsfont{\Glsentrylongpl{#2}}#3}

```

LSxtr@p@longpl@

```

3620 \def\@GLSxtr@p@longpl@#1#2[#3]{%
3621   {\mfirstucMakeUppercase{\glsfont{\Glsentrylongpl{#2}}#3}}

```

xtr@p@acrshort@

```

3622 \def\@glxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}

```

xtr@p@acrshort@

```

3623 \def\@Glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}

```

xtr@p@acrshort@

```

3624 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
3625   {\mfirstucMakeUppercase{\acronymfont{\Glsentryshort{#2}}#3}}

```

r@p@acrshortpl@

```

3626 \def\@glxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}

```

r@p@acrshortpl@

```

3627 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}

```

```

r@p@acrshortpl@
3628 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
3629   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}#3}}}}

```

```

sxtr@p@acrlong@
3630 \def\@glsxtr@p@acrlong@#1#2[#3]{\{\glsentrylong{#2}#3}}

```

```

sxtr@p@acrlong@
3631 \def\@Glsxtr@p@acrlong@#1#2[#3]{\{Glsentrylong{#2}#3}}

```

```

Sxtr@p@acrlong@
3632 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
3633   {\mfirstucMakeUppercase{\glsentrylong{#2}#3}}}}

```

```

tr@p@acrlongpl@
3634 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\{\glsentrylongpl{#2}#3}}

```

```

tr@p@acrlongpl@
3635 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\{Glsentrylongpl{#2}#3}}

```

```

tr@p@acrlongpl@
3636 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
3637   {\mfirstucMakeUppercase{\glsentrylongpl{#2}#3}}}}

```

Commands to minimise conflict.

```

\@glsxtrp@opt
3638 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

```

```

\glsxtrsetpopts   Used in glossary to switch hyperlinks on for the \glsxtrp type of commands.
3639 \newcommand*{\glsxtrsetpopts}[1]{%
3640   \renewcommand*{\@glsxtrp@opt}{#1}%
3641 }

```

```

\glossxtrsetpopts  Used in glossary to switch hyperlinks on for the \glsxtrp type of commands.
3642 \newcommand*{\glossxtrsetpopts}{%
3643   \glsxtrsetpopts{noindex}%
3644 }

```

```

\@@glsxtrp
3645 \newrobustcmd*{\@@glsxtrp}[2]{%
  Add scope.
3646   {%
3647     \let\glspostlinkhook\relax
3648     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
3649   }%
3650 }

```

\@glsxtrp

```
3651 \newrobustcmd*{\@glsxtrp}[2]{%
3652   \ifcsdef{gls#1}%
3653   {%
3654     \@glsxtrp{gls#1}{#2}%
3655   }%
3656   {%
3657     \ifcsdef{glsxtr#1}%
3658     {%
3659       \@glsxtrp{glsxtr#1}{#2}%
3660     }%
3661     {%
3662       \PackageError{glossaries-extra}{‘#1’ not recognised by
3663         \string\glsxtrp}{}%
3664     }%
3665   }%
3666 }
```

\@Glsxtrp

```
3667 \newrobustcmd*{\@Glsxtrp}[2]{%
3668   \ifcsdef{Gls#1}%
3669   {%
3670     \@glsxtrp{Gls#1}{#2}%
3671   }%
3672   {%
3673     \ifcsdef{Glsxtr#1}%
3674     {%
3675       \@glsxtrp{Glsxtr#1}{#2}%
3676     }%
3677     {%
3678       \PackageError{glossaries-extra}{‘#1’ not recognised by
3679         \string\Glsxtrp}{}%
3680     }%
3681   }%
3682 }
```

\@GLSxtrp

```
3683 \newrobustcmd*{\@GLSxtrp}[2]{%
3684   \ifcsdef{GLS#1}%
3685   {%
3686     \@glsxtrp{GLS#1}{#2}%
3687   }%
3688   {%
3689     \ifcsdef{GLSxtr#1}%
3690     {%
3691       \@glsxtrp{GLSxtr#1}{#2}%
3692     }%
3693     {%
3694       \PackageError{glossaries-extra}{‘#1’ not recognised by
```

```

3695      \string\GLSxtrp}{}%
3696    }%
3697  }%
3698 }

```

\glsxtr@entry@p

```

3699 \newrobustcmd*{\glsxtr@headentry@p}[2]{%
3700   \glsifattribute{#1}{headuc}{true}%
3701   {%
3702     \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
3703   }%
3704   {%
3705     \@gls@entry@field{#1}{#2}%
3706   }%
3707 }

```

\glsxtrp Not robust as it needs to expand somewhat.

```

3708 \ifdef\teorpdfstring
3709 {
3710   \newcommand{\glsxtrp}[2]{%
3711     \protect\NoCaseChange
3712     {%
3713       \protect\teorpdfstring
3714       {%
3715         \protect\glsxtrifinmark
3716         {%
3717           \ifcsdef{glsxtrhead#1}%
3718             {%
3719               {\protect\csuse{glsxtrhead#1}{#2}}%
3720             }%
3721             {%
3722               \glsxtr@headentry@p{#2}{#1}%
3723             }%
3724           }%
3725           {%
3726             \@glsxtrp{#1}{#2}%
3727           }%
3728         }%
3729         {%
3730           \protect\@gls@entry@field{#2}{#1}%
3731         }%
3732       }%
3733     }
3734 }
3735 {
3736   \newcommand{\glsxtrp}[2]{%
3737     \protect\NoCaseChange
3738     {%
3739       \protect\glsxtrifinmark

```

```

3740      {%
3741      \ifcsdef{glxtrhead#1}%
3742      {%
3743      {\protect\csuse{glxtrhead#1}}%
3744      }%
3745      {%
3746      \glxtr@headentry@p{#2}{#1}%
3747      }%
3748      }%
3749      {%
3750      \@glxtrp{#1}{#2}%
3751      }%
3752      }%
3753  }
3754 }

```

Provide short synonyms for the most common option.

`\glsp`

```

3755 \newcommand*{\glsp}{\glxtrp{short}}

```

`\glsp`

```

3756 \newcommand*{\glsp}{\glxtrp{text}}

```

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

```

3757 \ifdef\texorpdfstring
3758 {
3759   \newcommand{\Glsxtrp}[2]{%
3760     \protect\NoCaseChange
3761     {%
3762       \protect\texorpdfstring
3763       {%
3764         \protect\glxtrifinmark
3765         {%
3766           \ifcsdef{Glsxtrhead#1}%
3767           {%
3768             {\protect\csuse{Glsxtrhead#1}{#2}}%
3769             }%
3770             {%
3771               \protect\@Gls@entry@field{#2}{#1}%
3772               }%
3773             }%
3774             {%
3775               \@Glsxtrp{#1}{#2}%
3776             }%
3777             }%
3778             {%
3779               \protect\@Gls@entry@field{#2}{#1}%

```

```

3780     }%
3781   }%
3782 }
3783 }
3784 {
3785   \newcommand{\Glsxtrp}[2]{%
3786     \protect\NoCaseChange
3787     {%
3788       \protect\glsxtrifinmark
3789       {%
3790         \ifcsdef{Glsxtrhead#1}%
3791         {%
3792           {\protect\csuse{Glsxtrhead#1}}%
3793         }%
3794         {%
3795           \protect\@Gls@entry@field{#2}{#1}%
3796         }%
3797       }%
3798       {%
3799         \@Glsxtrp{#1}{#2}%
3800       }%
3801     }%
3802   }
3803 }

```

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

```

3804 \ifdef\texorpdfstring
3805 {
3806   \newcommand{\Glsxtrp}[2]{%
3807     \protect\NoCaseChange
3808     {%
3809       \protect\texorpdfstring
3810       {%
3811         \protect\glsxtrifinmark
3812         {%
3813           \ifcsdef{Glsxtr#1}%
3814           {%
3815             {\protect\Glsxtrshort[noindex,hyper=false]{#1}[]}%
3816           }%
3817           {%
3818             \protect\mfirstucMakeUppercase
3819             {%
3820               \protect\@gls@entry@field{#2}{#1}%
3821             }%
3822           }%
3823         }%
3824         {%
3825           \@Glsxtrp{#1}{#2}%
3826         }%

```

```

3827     }%
3828     {%
3829         \protect\@gls@entry@field{#2}{#1}%
3830     }%
3831 }%
3832 }
3833 }
3834 {
3835     \newcommand{\GLSxtrp}[2]{%
3836         \protect\NoCaseChange
3837         {%
3838             \protect\glsxtrifinmark
3839             {%
3840                 \ifcsdef{GLSxtr#1}%
3841                 {%
3842                     {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
3843                 }%
3844                 {%
3845                     \protect\mfirstucMakeUppercase
3846                     {%
3847                         \protect\@gls@entry@field{#2}{#1}%
3848                     }%
3849                 }%
3850             }%
3851             {%
3852                 \@GLSxtrp{#1}{#2}%
3853             }%
3854         }%
3855     }
3856 }

```

1.3.5 Entry Counting

The (use) entry counting mechanism from glossaries is adjusted here to work with category attributes. Provide a convenient command to enable entry counting, set the entrycount attribute for given categories and redefine \gls etc to use \cgl instead. This form of entry counting is provided to adjust the formatting if the number of times an entry has been used (through commands that unset the first use flag) doesn't exceeding the specified threshold. For link counting, see Section 1.4.

First adjust definitions of the unset and reset commands to provide a hook, but changing the flag can cause problems in certain situations, so to allow the normal unsetting to be temporarily disabled, \@glsunset is let to \@glsxtr@unset, which performs the actual unsetting through \@@glsunset and then does the hook. This means that the unsetting (and the hook) can switched off by redefining \@glsunset and then switched back on again by changing the definition back to \@glsxtr@unset.

```

\@glsxtr@unset Global unset.
3857 \newcommand*{\@glsxtr@unset}[1]{%

```

```

3858 \@@glsunset{#1}%
3859 \glsxtrpostunset{#1}%
3860 }%

\@glsunset Global unset.
3861 \let\@glsunset\@glsxtr@unset

glsxtrpostunset
3862 \newcommand*{\glsxtrpostunset}[1]{%

Provide a command to store a list of labels that will need unsetting.

tUnsetBuffering
3863 \newcommand*{\GlsXtrStartUnsetBuffering}{%
3864 \ifstar\s@GlsXtrStartUnsetBuffering\@GlsXtrStartUnsetBuffering
3865 }

tUnsetBuffering Unstarred version doesn't check for duplicates.
3866 \newcommand*{\@GlsXtrStartUnsetBuffering}{%
3867 \let\@glsxtr@org@unset@buffer\@glsxtr@unset@buffer
3868 \def\@glsxtr@unset@buffer{}%
3869 \let\@glsunset\@glsxtrbuffer@unset
3870 }

tUnsetBuffering Starred version checks for duplicates.
3871 \newcommand*{\s@GlsXtrStartUnsetBuffering}{%
3872 \let\@glsxtr@org@unset@buffer\@glsxtr@unset@buffer
3873 \def\@glsxtr@unset@buffer{}%
3874 \let\@glsunset\@glsxtrbuffer@nodup@unset
3875 }

xtrbuffer@unset This must use a global change since \gls may have to be placed inside \mbox (for example,
with soul commands).
3876 \newcommand*{\@glsxtrbuffer@unset}[1]{%
3877 \listxadd\@glsxtr@unset@buffer{#1}%
3878 }

fer@nodup@unset Alternative version that avoids duplicates. One level of expansion is performed on the ar-
gument in case it's a control sequence containing the label. (Not using \xifinlist as the
added complexity might cause problems that the buffering is trying to overcome.)
3879 \newcommand*{\@glsxtrbuffer@nodup@unset}[1]{%
3880 \expandafter\ifinlist\expandafter{#1}{\@glsxtr@unset@buffer}{}%
3881 {\listxadd\@glsxtr@unset@buffer{#1}}%
3882 }

pUnsetBuffering
3883 \newcommand*{\GlsXtrStopUnsetBuffering}{%
3884 \ifstar\s@GlsXtrStopUnsetBuffering\@GlsXtrStopUnsetBuffering
3885 }

```


pUnsetBuffering Unstarred form (global unset).

```
3886 \newcommand*{\@GlsXtrStopUnsetBuffering}{%
3887   \let\@glsunset\@glxtr@unset
3888   \forlistloop\@glsunset\@glxtr@unset@buffer
3889   \let\@glxtr@unset@buffer\@glxtr@org@unset@buffer
3890 }
```

pUnsetBuffering Starred form (local unset).

```
3891 \newcommand*{\s@GlsXtrStopUnsetBuffering}{%
3892   \forlistloop\@glslocalunset\@glxtr@unset@buffer
3893   \let\@glsunset\@glxtr@unset
3894 }
```

dUnsetBuffering Discards pending buffer and restores \glsunset.

```
3895 \newcommand*{\GlsXtrDiscardUnsetBuffering}{%
3896   \let\@glsunset\@glxtr@unset
3897   \let\@glxtr@unset@buffer\@glxtr@org@unset@buffer
3898 }
```

setBufferedList Iterate over labels stored in the current buffer. The argument is the handler macro.

```
3899 \newcommand*{\GlsXtrForUnsetBufferedList}[1]{%
3900   \forlistloop#1\@glxtr@unset@buffer
3901 }
```

\@glslocalunset Local unset.

```
3902 \renewcommand*{\@glslocalunset}[1]{%
3903   \@glslocalunset{#1}%
3904   \glxtrpostlocalunset{#1}%
3905 }
```

rpostlocalunset

```
3906 \newcommand*{\glxtrpostlocalunset}[1]{}
```

\@glsreset Global reset.

```
3907 \renewcommand*{\@glsreset}[1]{%
3908   \@glsreset{#1}%
3909   \glxtrpostreset{#1}%
3910 }
```

glxtrpostreset

```
3911 \newcommand*{\glxtrpostreset}[1]{}
```

\@glslocalreset Local reset.

```
3912 \renewcommand*{\@glslocalreset}[1]{%
3913   \@glslocalreset{#1}%
3914   \glxtrpostlocalreset{#1}%
3915 }
```

rpostlocalreset

```
3916 \newcommand*{\glxtrpostlocalreset}[1]{}
```

slocalreseteach Locally reset a list of entries.

```
3917 \newcommand*{\glslocalreseteach}[1]{%
3918   \gls@ifnotmeasuring
3919   {%
3920     \@for\@gls@thislabel:=#1\do{%
3921       \glsdoifexists{\@gls@thislabel}%
3922       {%
3923         \glslocalreset{\@gls@thislabel}%
3924       }%
3925     }%
3926   }%
3927 }
```

slocalunseteach Locally unset a list of entries.

```
3928 \newcommand*{\glslocalunseteach}[1]{%
3929   \gls@ifnotmeasuring
3930   {%
3931     \@for\@gls@thislabel:=#1\do{%
3932       \glsdoifexists{\@gls@thislabel}%
3933       {%
3934         \glslocalunset{\@gls@thislabel}%
3935       }%
3936     }%
3937   }%
3938 }
```

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

```
3939 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

 Enable entry counting:

```
3940   \glsenableentrycount
```

 Redefine \gls etc:

```
3941   \renewcommand*{\gls}{\cglss}%
3942   \renewcommand*{\Gls}{\cGls}%
3943   \renewcommand*{\glspl}{\cglspl}%
3944   \renewcommand*{\Glspl}{\cGlspl}%
3945   \renewcommand*{\GLS}{\cGLS}%
3946   \renewcommand*{\GLSpl}{\cGLSpl}%
```

 Set the entrycount attribute:

```
3947   \@glxtr@setentrycountunsetattr{#1}{#2}%
```

 In case this command is used again:

```
3948   \let\GlsXtrEnableEntryCounting\@glxtr@setentrycountunsetattr
3949   \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
```

```

3950 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
3951   can't be used with \string\GlsXtrEnableEntryCounting}%
3952 {Use one or other but not both commands}}%
3953 }

```

ycountunsetattr

```

3954 \newcommand*{\@glxtr@setentrycountunsetattr}[2]{%
3955   \@for\@glxtr@cat:=#1\do
3956   {%
3957     \ifdefempty{\@glxtr@cat}{}%
3958     {%
3959       \glsssetcategoryattribute{\@glxtr@cat}{entrycount}{#2}%
3960     }%
3961   }%
3962 }

```

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

nableentrycount

```

3963 \renewcommand*{\glsenableentrycount}{%

```

Enable new fields:

```

3964 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%

```

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

```

3965 \renewcommand*{\gls@defdocnewglossaryentry}{%
3966   \renewcommand*{\newglossaryentry}[2]{%
3967     \PackageError{glossaries}{\string\newglossaryentry\space
3968       may only be used in the preamble when entry counting has
3969       been activated}{If you use \string\glsenableentrycount\space
3970       you must place all entry definitions in the preamble not in
3971       the document environment}%
3972   }%
3973 }%

```

New commands to access new fields:

```

3974 \newcommand*{\glsentrycurrcount}[1]{%
3975   \ifcsundef{glo@\glsdetoklabel{##1}@currcount}%
3976   {0}{\@gls@entry@field{##1}{currcount}}%
3977 }%
3978 \newcommand*{\glsentryprevcount}[1]{%
3979   \ifcsundef{glo@\glsdetoklabel{##1}@prevcount}%
3980   {0}{\@gls@entry@field{##1}{prevcount}}%
3981 }%

```

Adjust post unset and reset:

```

3982 \let\@glxtr@entrycount@org@unset\glxtrpostunset
3983 \renewcommand*{\glxtrpostunset}[1]{%
3984   \@glxtr@entrycount@org@unset{##1}%
3985   \@gls@increment@currcount{##1}%
3986 }%

```

```

3987 \let\@glxtr@entrycount@org@localunset\glxtrpostlocalunset
3988 \renewcommand*{\glxtrpostlocalunset}[1]{%
3989   \@glxtr@entrycount@org@localunset{##1}%
3990   \@gls@local@increment@currcount{##1}%
3991 }%
3992 \let\@glxtr@entrycount@org@reset\glxtrpostreset
3993 \renewcommand*{\glxtrpostreset}[1]{%
3994   \@glxtr@entrycount@org@reset{##1}%
3995   \csgdef{glo@glsdetoklabel{##1}@currcount}{0}%
3996 }%
3997 \let\@glxtr@entrycount@org@localreset\glxtrpostlocalreset
3998 \renewcommand*{\glxtrpostlocalreset}[1]{%
3999   \@glxtr@entrycount@org@localreset{##1}%
4000   \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
4001 }%

```

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

```

4002 \let\@cgl@\@cgl@
4003 \let\@cgl@pl\@cgl@pl@

4004 \let\@cGls@\@cGls@
4005 \let\@cGls@pl\@cGls@pl@
4006 \let\@cGLS@\@cGLS@
4007 \let\@cGLS@pl\@cGLS@pl@

```

The rest is as the original definition.

```

4008 \AtEndDocument{\@gls@write@entrycounts}%
4009 \renewcommand*{\@gls@entry@count}[2]{%
4010   \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
4011 }%
4012 \let\glsenableentrycount\relax
4013 \renewcommand*{\glsenableentryunitcount}{%
4014   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space
4015     can't be used with \string\glsenableentrycount}%
4016   {Use one or other but not both commands}%
4017 }%
4018 }

```

`\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.

```

4019 \renewcommand*{\@gls@write@entrycounts}{%
4020   \immediate\write\@auxout
4021     {\string\providecommand*{\string\@gls@entry@count}[2]{}}%
4022   \count@=0\relax
4023   \forallglsentries{\@glsentry}{%
4024     \gls@hasattribute{\@glsentry}{entrycount}%
4025     {%
4026       \ifglsused{\@glsentry}%
4027       {%

```

```

4028      \immediate\write\@auxout
4029      {\string\@gls@entry@count{\@glsentry}{\@glsentrycurrcount{\@glsentry}}}%
4030      }%
4031      {}%
4032      \advance\count@ by \@ne
4033      }%
4034      {}%
4035      }%
4036      \ifnum\count@=0
4037      \GlossariesExtraWarningNoLine{Entry counting has been enabled
4038      \MessageBreak with \string\glsenableentrycount\space but the
4039      \MessageBreak attribute ‘entrycount’ hasn’t
4040      \MessageBreak been assigned to any of the defined
4041      \MessageBreak entries}%
4042      \fi
4043 }

```

rifcounttrigger

<code>\glxtrifcounttrigger{<label>}{<trigger format>}{<normal>}</code>
--

```

4044 \newcommand*{\glxtrifcounttrigger}[3]{%
4045   \glshasattribute{#1}{entrycount}%
4046   {%
4047     \ifnum\glsentryprevcount{#1}>\glsgetattribute{#1}{entrycount}\relax
4048     #3%
4049     \else
4050     #2%
4051     \fi
4052   }%
4053   {#3}%
4054 }

```

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

\@@cgl@

```

4055 \def\@@cgl@#1#2[#3]{%
4056   \glxtrifcounttrigger{#2}%
4057   {%
4058     \cglformat{#2}{#3}%
4059     \glset{#2}%
4060   }%
4061   {%
4062     \@gls@{#1}{#2}[#3]%
4063   }%
4064 }%

```

\@@cglsp1@

```
4065 \def\@@cglsp1@#1#2[#3]{%
4066   \glxtrifcounttrigger{#2}%
4067   {%
4068     \cglsp1format{#2}{#3}%
4069     \glset{#2}%
4070   }%
4071   {%
4072     \@glsp1@{#1}{#2}[#3]%
4073   }%
4074 }%
```

\@@cGls@

```
4075 \def\@@cGls@#1#2[#3]{%
4076   \glxtrifcounttrigger{#2}%
4077   {%
4078     \cGlsformat{#2}{#3}%
4079     \glset{#2}%
4080   }%
4081   {%
4082     \@Gls@{#1}{#2}[#3]%
4083   }%
4084 }%
```

\@@cGlsp1@

```
4085 \def\@@cGlsp1@#1#2[#3]{%
4086   \glxtrifcounttrigger{#2}%
4087   {%
4088     \cGlsp1format{#2}{#3}%
4089     \glset{#2}%
4090   }%
4091   {%
4092     \@Glsp1@{#1}{#2}[#3]%
4093   }%
4094 }%
```

\@@cGLS@

```
4095 \def\@@cGLS@#1#2[#3]{%
4096   \glxtrifcounttrigger{#2}%
4097   {%
4098     \cGLSformat{#2}{#3}%
4099     \glset{#2}%
4100   }%
4101   {%
4102     \@GLS@{#1}{#2}[#3]%
4103   }%
4104 }%
```

\@@cGLSp1@

```

4105 \def\@cGLSpl@#1#2[#3]{%
4106   \glstrifcounttrigger{#2}%
4107   {%
4108     \cGLSplformat{#2}{#3}%
4109     \glset{#2}%
4110   }%
4111   {%
4112     \@GLSpl@{#1}{#2}[#3]%
4113   }%
4114 }%

```

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

\@cgl's@

```
4115 \def\@cgl's@#1#2[#3]{\@gl's@{#1}{#2}[#3]}
```

\@cGL's@

```
4116 \def\@cGL's@#1#2[#3]{\@GL's@{#1}{#2}[#3]}
```

\@cgl'spl@

```
4117 \def\@cgl'spl@#1#2[#3]{\@gl'spl@{#1}{#2}[#3]}
```

\@cGL'spl@

```
4118 \def\@cGL'spl@#1#2[#3]{\@GL'spl@{#1}{#2}[#3]}
```

Add all upper case versions not provided by glossaries.

\cGLS

```
4119 \newrobustcmd*{\cGLS}{\@gl's@hyp@opt\@cGLS}
```

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

```

4120 \newcommand*{\@cGLS}[2][ ]{%
4121   \new@ifnextchar[{\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2}[ ]}%
4122 }

```

\@cGLS@

```
4123 \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.

```

4124 \newcommand*{\cGLSformat}[2]{%
4125   \expandafter\mfirstucMakeUppercase\expandafter{\cgl'sformat{#1}{#2}}%
4126 }

```

\cGLSpl

```
4127 \newrobustcmd*{\cGLSpl}{\@gl's@hyp@opt\@cGLSpl}
```

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

```
4128 \newcommand*{\@cGLSp1}[2] [] {%
4129   \new@ifnextchar[{\@cGLSp1@{#1}{#2}}{\@cGLSp1@{#1}{#2} []}%
4130 }
```

\@cGLSp1@

```
4131 \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.

```
4132 \newcommand*{\cGLSplformat}[2] {%
4133   \expandafter\mfirstuc\MakeUppercase\expandafter{\cglSplformat{#1}{#2}}%
4134 }
```

Modify the trigger formats to check for the regular attribute.

\cglSformat

```
4135 \renewcommand*{\cglSformat}[2] {%
4136   \glsifregular{#1}
4137   {\glsentryfirst{#1}}%
4138   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
4139 }
```

\cGlsformat

```
4140 \renewcommand*{\cGlsformat}[2] {%
4141   \glsifregular{#1}
4142   {\Glsentryfirst{#1}}%
4143   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
4144 }
```

\cglSplformat

```
4145 \renewcommand*{\cglSplformat}[2] {%
4146   \glsifregular{#1}
4147   {\glsentryfirstplural{#1}}%
4148   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
4149 }
```

\cGlsplformat

```
4150 \renewcommand*{\cGlsplformat}[2] {%
4151   \glsifregular{#1}
4152   {\Glsentryfirstplural{#1}}%
4153   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}}%#2%
4154 }
```

New code similar to above for unit counting.

defunitcounters

```
4155 \newcommand*{\@@newglossaryentry@defunitcounters}{%
```



```

4156 \protected@edef\@glo@countunit{\csuse{@glxtr@categoryattr@@\@glo@category @unitcount}}}%
4157 \ifvoid\@glo@countunit
4158 {}%
4159 {%
4160 \@glxtr@ifunitcounter{\@glo@countunit}%
4161 {}%
4162 {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}%
4163 }%
4164 }

```

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

```

4165 \newcommand*\@glxtr@unitcountlist{}

```

@addunitcounter

```

4166 \newcommand*\@glxtr@addunitcounter}[1]{%
4167 \listadd{\@glxtr@unitcountlist}{#1}%
4168 \ifcsundef{glxtr@theunit@#1}
4169 {%
4170 \ifcsdef{theH#1}%
4171 {\csdef{glxtr@theunit@#1}{\csuse{theH#1}}}%
4172 {\csdef{glxtr@theunit@#1}{\csuse{the#1}}}%
4173 }%
4174 {}%
4175 }

```

r@ifunitcounter

```

4176 \newcommand*\@glxtr@ifunitcounter}[3]{%
4177 \xifinlist{#1}{\@glxtr@unitcountlist}{#2}{#3}%
4178 }

```

urrentunitcount

```

4179 \newcommand*\@glxtr@currentunitcount[1]{%
4180 glo@\glstoklabel{#1}@currunit@\glsggetattribute{#1}{unitcount}.%
4181 \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}%
4182 }

```

eviousunitcount

```

4183 \newcommand*\@glxtr@previousunitcount[1]{%
4184 glo@\glstoklabel{#1}@prevunit@\glsggetattribute{#1}{unitcount}.%
4185 \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}%
4186 }

```

t@currunitcount

```

4187 \newcommand*\@glxtr@increment@currunitcount[1]{%
4188 \glshasattribute{#1}{unitcount}%
4189 {%
4190 \protected@edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
4191 \ifcsundef{\@glxtr@csname}%

```

```

4192   {%
4193     \csgdef{\@glsxtr@csname}{1}%
4194     \listcsxadd
4195       {glo@\glsdetoklabel{#1}@unitlist}%
4196       {\glsgetattribute{#1}{unitcount}.%
4197       \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
4198     }%
4199   }%
4200   {%
4201     \csxdef{\@glsxtr@csname}%
4202     {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
4203   }%
4204   }%
4205   {}%
4206 }

```

t@currunitcount

```

4207 \newcommand*{\@gls@local@increment@currunitcount}[1]{%
4208   \gls@hasattribute{#1}{unitcount}%
4209   {%
4210     \protected@edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
4211     \ifcsundef{\@glsxtr@csname}%
4212     {%
4213       \csdef{\@glsxtr@csname}{1}%
4214       \listcseadd
4215         {glo@\glsdetoklabel{#1}@unitlist}%
4216         {\glsgetattribute{#1}{unitcount}.%
4217         \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
4218       }%
4219     }%
4220     {%
4221       \csedef{\@glsxtr@csname}%
4222       {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
4223     }%
4224   }%
4225   {}%
4226 }

```

r@currunitcount

```

4227 \newcommand*{\@glsxtr@currunitcount}[2]{%
4228   \ifcsundef
4229     {glo@\glsdetoklabel{#1}@currunit@#2}%
4230     {0}%
4231     {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}%
4232 }%

```

r@prevunitcount

```

4233 \newcommand*{\@glsxtr@prevunitcount}[2]{%
4234   \ifcsundef

```

```

4235 {glo@\glsdetoklabel{#1}@prevunit@#2}%
4236 {0}%
4237 {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}%
4238 }%

```

entryunitcount

```

4239 \newcommand*\glsenableentryunitcount{%
  Enable new fields:
4240 \appto@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
4241 \renewcommand*\gls@defdocnewglossaryentry{%
4242 \renewcommand*\newglossaryentry[2]{%
4243 \PackageError{glossaries}{\string\newglossaryentry\space
4244 may only be used in the preamble when entry counting has
4245 been activated}{If you use \string\glsenableentryunitcount\space
4246 you must place all entry definitions in the preamble not in
4247 the document environment}%
4248 }%
4249 }%

```

New commands to access new fields:

```

4250 \newcommand*\glsentrycurrcount}[1]{%
4251 \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
4252 \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
4253 }%
4254 \newcommand*\glsentryprevcount}[1]{%
4255 \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
4256 \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
4257 }%

```

Access total count:

```

4258 \newcommand*\glsentryprevtotalcount}[1]{%
4259 \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
4260 {0}%
4261 {%
4262 \number\csuse{glo@\glsdetoklabel{##1}@prevunittotal}
4263 }%
4264 }%

```

Access max value:

```

4265 \newcommand*\glsentryprevmaxcount}[1]{%
4266 \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
4267 {0}%
4268 {%
4269 \number\csuse{glo@\glsdetoklabel{##1}@prevunitmax}
4270 }%
4271 }%

```

Adjust post unset and reset:

```

4272 \let@glsxtr@entryunitcount@org@unset\glsxtrpostunset

```

```

4273 \renewcommand*{\glxtrpostunset}[1]{%
4274   \@glxtr@entryunitcount@org@unset{##1}%
4275   \@glx@increment@currunitcount{##1}%
4276 }%
4277 \let\@glxtr@entryunitcount@org@localunset\glxtrpostlocalunset
4278 \renewcommand*{\glxtrpostlocalunset}[1]{%
4279   \@glxtr@entryunitcount@org@localunset{##1}%
4280   \@glx@local@increment@currunitcount{##1}%
4281 }%
4282 \let\@glxtr@entryunitcount@org@reset\glxtrpostreset
4283 \renewcommand*{\glxtrpostreset}[1]{%
4284   \glshasattribute{##1}{unitcount}%
4285   {%
4286     \protected@edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
4287     \ifcsundef{\@glxtr@csname}%
4288     {}%
4289     {\csgdef{\@glxtr@csname}{0}}%
4290   }%
4291   {}%
4292 }%
4293 \let\@glxtr@entryunitcount@org@localreset\glxtrpostlocalreset
4294 \renewcommand*{\glxtrpostlocalreset}[1]{%
4295   \@glxtr@entryunitcount@org@localreset{##1}%
4296   \glshasattribute{##1}{unitcount}%
4297   {%
4298     \protected@edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
4299     \ifcsundef{\@glxtr@csname}%
4300     {}%
4301     {\csdef{\@glxtr@csname}{0}}%
4302   }%
4303   {}%
4304 }%

```

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

```

4305 \let\@cgl@\@cgl@
4306 \let\@cgl@pl@\@cgl@pl@
4307 \let\@cGl@\@cGl@
4308 \let\@cGl@pl@\@cGl@pl@
4309 \let\@cGLS@\@cGLS@
4310 \let\@cGL@pl@\@cGL@pl@

```

Write information to the aux file.

```

4311 \AtEndDocument{\@glx@write@entryunitcounts}%
4312 \renewcommand*{\@glx@entry@unitcount}[3]{%
4313   \csgdef{glo@\glx@detoklabel{##1}@prevunit@##3}{##2}%
4314   \ifcsundef{glo@\glx@detoklabel{##1}@prevunittotal}%

```

```

4315   {\csgdef{glo@\glsdetoklabel{##1}@prevunittotal}{##2}}%
4316   {%
4317     \csxdef{glo@\glsdetoklabel{##1}@prevunittotal}{
4318       \number\numexpr\csuse{glo@\glsdetoklabel{##1}@prevunittotal}+##2}%
4319     }%
4320     \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
4321     {\csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}}%
4322     {%
4323       \ifnum\csuse{glo@\glsdetoklabel{##1}@prevunitmax}<##2
4324       \csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}%
4325       \fi
4326     }%
4327   }%
4328   \let\glsenableentryunitcount\relax
4329   \renewcommand*{\glsenableentrycount}{%
4330     \PackageError{glossaries-extra}{\string\glsenableentrycount\space
4331       can't be used with \string\glsenableentryunitcount}%
4332     {Use one or other but not both commands}%
4333   }%
4334 }
4335 \onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

4336 \newcommand*{\@gls@entry@unitcount}[3]{%

```

ryunitcounts@do

```

4337 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
4338   \immediate\write\@auxout
4339   {\string\@gls@entry@unitcount
4340     {\@glsentry}%
4341     {\@glsxtr@currunitcount{\@glsentry}{#1}%
4342     }%
4343     {#1}}%
4344 }

```

entryunitcounts

```

4345 \newcommand*{\@gls@write@entryunitcounts}{%
4346   \immediate\write\@auxout
4347   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
4348   \count@=0\relax
4349   \forallglsentries{\@glsentry}{%
4350     \gls@hasattribute{\@glsentry}{unitcount}%
4351     {%
4352       \ifglsused{\@glsentry}%
4353       {%
4354         \forlistcsloop
4355         {\@gls@write@entryunitcounts@do}%
4356         {glo@\glsdetoklabel{\@glsentry}@unitlist}%
4357       }%

```

```

4358     {}%
4359     \advance\count@ by \@ne
4360   }%
4361   {}%
4362 }%
4363 \ifnum\count@=0
4364   \GlossariesExtraWarningNoLine{Entry counting has been enabled
4365     \MessageBreak with \string\glsenableentryunitcount\space but the
4366     \MessageBreak attribute ‘unitcount’ hasn’t
4367     \MessageBreak been assigned to any of the defined
4368     \MessageBreak entries}%
4369 \fi
4370 }

```

`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.

```

4371 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

4372 \glsenableentryunitcount

```

Redefine `\gls` etc:

```

4373 \renewcommand*{\gls}{\cgl}%
4374 \renewcommand*{\Gls}{\cGls}%
4375 \renewcommand*{\glspl}{\cglspl}%
4376 \renewcommand*{\Glspl}{\cGlspl}%
4377 \renewcommand*{\GLS}{\cGLS}%
4378 \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

4379 \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

4380 \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
4381 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
4382   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
4383     can’t be used with \string\GlsXtrEnableEntryUnitCounting}%
4384   {Use one or other but not both commands}}%
4385 }

```

`tcountunsetattr`

```

4386 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
4387   \@for\@glsxtr@cat:=#1\do
4388   {%
4389     \ifdefempty{\@glsxtr@cat}{}%
4390     {%
4391       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
4392       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
4393     }%
4394   }%
4395 }

```

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`

```
4396 \renewcommand*{\SetGenericNewAcronym}{%
    Make sure \RestoreAcronyms has been used.
4397 \ifdefequal\@addtoacronymlists\@glstr@org@addtoacronymlists
4398 {}%
4399 {%
4400 \GlossariesWarning{\string\SetGenericNewAcronym\space used
4401 without restoring base acronym functions with
4402 \string\RestoreAcronyms}%
4403 }%
4404 \let\@Gls@entryname\@Gls@acentryname

    Redefine \newacronym:
4405 \renewcommand{\newacronym}[4][{}]{%
4406 \ifdefempty{\@glssacronymlists}%
4407 {%
4408 \def\@glo@type{\acronymtype}%
4409 \setkeys{glossentry}{##1}%
4410 \DeclareAcronymList{\@glo@type}%
4411 }%
4412 {}%
4413 \glsskeylisttok{##1}%
4414 \glsslabeltok{##2}%
4415 \glssshorttok{##3}%
4416 \glsslongtok{##4}%
4417 \newacronymhook
4418 \protected@edef\@do@newglossaryentry{%
4419 \noexpand\newglossaryentry{\the\glsslabeltok}%
4420 {%
4421 type=\acronymtype,%
4422 name={\expandonce{\acronymentry{##2}}},%
4423 sort={\acronymssort{\the\glssshorttok}{\the\glsslongtok}},%
4424 text={\the\glssshorttok},%
4425 short={\the\glssshorttok},%
4426 shortplural={\the\glssshorttok\noexpand\acrpluralsuffix},%
4427 long={\the\glsslongtok},%
4428 longplural={\the\glsslongtok\noexpand\acrpluralsuffix},%
4429 category=acronym,
4430 \GenericAcronymFields,%
4431 \the\glsskeylisttok
```

```

4432 }%
4433 }%
4434 \@do@newglossaryentry
4435 }%
4436 \renewcommand*{\acrfullfmt}[3]{%
4437   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
4438 \renewcommand*{\Acrfullfmt}[3]{%
4439   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
4440 \renewcommand*{\ACRfullfmt}[3]{%
4441   \glslink[##1]{##2}{%
4442     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
4443 \renewcommand*{\acrfullplfmt}[3]{%
4444   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
4445 \renewcommand*{\Acrfullplfmt}[3]{%
4446   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
4447 \renewcommand*{\ACRfullplfmt}[3]{%
4448   \glslink[##1]{##2}{%
4449     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
4450 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}}%
4451 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
4452 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}}%
4453 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
4454 }

```

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:

```

4455 \let\@glxtr@org@setacronymstyle\setacronymstyle
4456 \let\@glxtr@org@newacronymstyle\newacronymstyle

```

Save the list of acronyms in case they are required.

`tr@acronymlists`

```

4457 \let\@glxtr@acronymlists\@glsacronymlists

```

`dtoacronymlists`

```

4458 \let\@glxtr@org@addtoacronymlists\@addtoacronymlists

```

`setacronymlists`

```

4459 \let\@glxtr@org@setacronymlists\SetAcronymLists

```

Need to provide a replacement for `\forallacronyms` since `\@glsacronymlists` isn't available.

`lsxtr@abbrlists`

```

4460 \newcommand{\@glxtr@abbrlists}{}

```

`breviationlists`

```

4461 \newcommand*{\forallabbreviationlists}[2]{%

```



```

4462 \for#1:=\@glxtr@abbrlists\do{\ifdefempty{#1}-{#2}}%
4463 }

```

bbreviationlist

```

4464 \newcommand*{\@glxtr@addabbreviationlist}[1]{%
4465   \protected@edef\@glo@type{#1}%
4466   \ifdefempty\@glxtr@abbrlists
4467   {\let\@glxtr@abbrlists\@glo@type}%
4468   {%
4469     \ifdefequal\@glxtr@abbrlists\@glo@type
4470     }%
4471     {%
4472       \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@abbrlists}{}%
4473       {\protected@eappto\@glxtr@abbrlists{,\@glo@type}}%
4474     }%
4475   }%
4476 }

```

\forallacronyms Modify to add warning.

```

4477 \renewcommand*{\forallacronyms}[2]{%
4478   \@glxtr@base@acrcmd\forallacronyms\forallabbreviationlists
4479   \@for#1:=\@glsacronymlists\do{\ifx#1\@empty\else#2\fi}%
4480 }

```

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

```

4481 \newcommand*{\MakeAcronymsAbbreviations}{%

```

Undo acronym display style:

```

4482 \@for\@gls@type:=\@glsacronymlists\do{%
4483   \csgdef{gls@\@gls@type @entryfmt}{\glsentryfmt}%
4484 }%

```

Save and clear acronym list.

```

4485 \let\@glxtr@acronymlists\@glsacronymlists
4486 \let\@glsacronymlists\@empty
4487 \let\@addtoacronymlists\@gobble
4488 \let\SetAcronymLists\@gobble

```

Warn if \acrshort etc are used.

```

4489 \let\@glxtr@base@acrcmd\@glxtr@base@acrcmd@warn

```

Redefine \newacronym to use same interface as \newabbreviation.

```

4490 \renewcommand*{\newacronym}[4][ ]{%
4491   \glxtr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
4492 }%
4493 \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
4494 \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%

```

```

4495 \renewcommand*\setacronymstyle}[1]{%
4496 \PackageError{glossaries-extra}{\string\setacronymstyle{##1}
4497 unavailable.
4498 Use \string\setabbreviationstyle[acronym]\space instead.
4499 The original acronym interface can be restored with
4500 \string\RestoreAcronyms}{}%
4501 }%
4502 \renewcommand*\newacronymstyle}[1]{%
4503 \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
4504 available unless you restore the original acronym interface with
4505 \string\RestoreAcronyms}%
4506 \@glxtr@org@newacronymstyle{##1}%
4507 }%
4508 }

```

Switch acronyms to abbreviations:

```
4509 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```
4510 \newcommand*\RestoreAcronyms{%
```

Restore acronym list.

```

4511 \let\@glsacronymlists\@glxtr@acronymlists
4512 \let\@addtoacronymlists\@glxtr@org@addtoacronymlists
4513 \let\SetAcronymLists\@glxtr@org@setacronymlists

```

Suppress warnings if \acrshort etc are used.

```
4514 \let\@glxtr@base@acrcmd\@gobbletwo
```

Restore acronym display style:

```

4515 \@for\@gls@type:=\@glsacronymlists\do{%
4516 \SetDefaultAcronymDisplayStyle{\@gls@type}%
4517 }%

```

Switch to the generic acronym mechanism.

```

4518 \SetGenericNewAcronym
4519 \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
4520 \renewcommand{\acronymfont}[1]{##1}%
4521 \let\setacronymstyle\@glxtr@org@setacronymstyle
4522 \let\newacronymstyle\@glxtr@org@newacronymstyle

```

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

```

4523 \renewcommand*\@gls@link@checkfirsthyper{%
4524 \ifglsused{glslabel}%
4525 {\let\glxtrifwasfirstuse\@secondoftwo}
4526 {\let\glxtrifwasfirstuse\@firstoftwo}%
4527 \@glxtr@org@checkfirsthyper
4528 }
4529 \glssetcategoryattribute{acronym}{regular}{false}%
4530 \setacronymstyle{long-short}%
4531 }

```

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

```
4532 \renewcommand*{\glsacspace}[1]{%
4533   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
4534   \ifdim\dimen@<\glsacspacemax~\else\space\fi
4535 }
```

`\glsacspacemax` Value used in the above.

```
4536 \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`

```
4537 \newcommand*{\@glxtr@reg@glosslist}{}
```

Save the original definition of `\makeglossaries`:

```
4538 \let\@glxtr@org@makeglossaries\makeglossaries
```

`noprintglossary` This command was only introduced to glossaries v4.47 so it may not be defined.

```
4539 \providecommand\@makeglossaries@warn@noprintglossary{%
4540   \ifdefstring{\@glo@types}{,}%
4541   {%
4542     \GlossariesWarningNoLine{No glossaries have been defined}%
4543   }%
4544   {%
4545     \GlossariesWarningNoLine{No \string\printglossary\space
4546       or \string\printglossaries\space
4547       found. ^^J(Remove \string\makeglossaries\space if you
4548       don't want any glossaries.) ^^JThis document will not
4549       have a glossary}%
4550   }%
4551 }
```

`omakeglossaries` glossaries v4.45 introduced `\@domakeglossaries` to provide a way of disabling `\makeglossaries`. If it hasn't been defined, define here to do its argument:

```
4552 \providecommand{\@domakeglossaries}[1]{#1}
```

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. The optional argument version shouldn't be used with record.

\makeglossaries

```

4553 \renewcommand*{\makeglossaries}[1] [] {%
4554   \@domakeglossaries
4555   {%
4556     \@glxtr@if@record@only
4557     {%
4558       \PackageError{glossaries-extra}{\string\makeglossaries\space
4559         not permitted\MessageBreak with record=\@glxtr@record@setting\space
4560         package option}%
4561       {You may only use \string\makeglossaries\space with
4562         record=off or record=hybrid options}%
4563     }%
4564     {%
4565       \ifblank{#1}%
4566       {%
4567         \@glxtr@org@makeglossaries

4568         \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
4569           \let\warn@noprintglossary\@glxtr@warn@hybrid@noprintgloss
4570         \fi
4571       }%
4572     }%
4573     \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
4574       \PackageError{glossaries-extra}{\string\makeglossaries[#1]\space
4575         not permitted\MessageBreak with record=\@glxtr@record@setting\space package option}%
4576       {You may only use the hybrid \string\makeglossaries[...]\space with
4577         record=off option}%
4578     \else

```

\@gls@@automake@immediate was introduced to glossaries v4.42 so it may not be defined.

```

4579     \ifdef\@gls@@automake@immediate{\@gls@@automake@immediate}{}%
4580     \protected@edef\@glxtr@reg@glosslist{#1}%
4581     \ifundef{\glswrite}{\newwrite\glswrite}{}%
4582     \protected@write\@auxout{}{\string\providecommand
4583       \string\@glsorder[1]{}}
4584     \protected@write\@auxout{}{\string\providecommand
4585       \string\@istfilename[1]{}}
4586     \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
4587     \protected@write\@auxout{}{\string\@glsorder{\glsorder}}
4588     \protected@write\@auxout{}{\string\glxtr@makeglossaries{#1}}
4589     \write\@auxout{\string\providecommand\string\@gls@reference[3]{}}%

```

Iterate through each supplied glossary type and activate it.

```

4590     \@for\@glo@type:=#1\do{%
4591       \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
4592     }%

```

New glossaries must be created before \makeglossaries:

```

4593     \renewcommand*\newglossary[4] [] {%
4594     \PackageError{glossaries}{New glossaries

```

```

4595      must be created before \string\makeglossaries}{You need
4596      to move \string\makeglossaries\space after all your
4597      \string\newglossary\space commands}}%

```

Any subsequence instances of this command should have no effect.

```

4598      \let\@makeglossary\@gobble

```

Version 1.42 removed letting \makeglossary to \relax (no kernel redefs may be in effect).

```

4599      \renewcommand\makeglossaries[1] [] {}%

```

Disable all commands that have no effect after \makeglossaries

```

4600      \@disable@onlypremakeg

```

Allow see key:

```

4601      \let\gls@checkseeallowed\relax

```

Adjust \@do@seeglossary. This needs to check for the entry's existence but don't increment associated counter.

```

4602      \renewcommand*\@do@seeglossary}[2] {%
4603          \glsdoifexists{##1}%
4604          {%
4605              \protected@edef\@gls@label{\glsdetoklabel{##1}}%
4606              \protected@edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
4607              \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
4608              {\@glsxtr@org@doseeglossary{##1}{##2}}%
4609              {%
4610                  \@glsxtrwrglossmark
4611                  \protected@write\@auxout{}{%
4612                      \string\@gls@reference
4613                      {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
4614                      }%
4615                      }%
4616                      }%
4617                      }%

```

Adjust \@do@wrglossary

```

4618      \let\@glsxtr@do@wrglossary\@do@wrglossary
4619      \def\@do@wrglossary{%
4620          \protected@edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
4621          \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
4622          {\@glsxtr@do@wrglossary}%
4623          {\@gls@noidxglossary}%
4624          }%

```

Suppress warning about no \makeglossaries

```

4625      \let\warn@nomakeglossaries\relax
4626      \let\warn@noprintglossary\@makeglossaries@warn@noprintglossary

```

Only warn for glossaries not listed.

```

4627      \renewcommand{\@gls@noref@warn}[1] {%

```

```

4628         \protected@edef\@gls@type{##1}%
4629         \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
4630     {%
4631         \GlossariesExtraWarning{Can't use
4632             \string\printnoidxglossary[type={\@gls@type}]
4633             when '\@gls@type' is listed in the optional argument of
4634             \string\makeglossaries}%
4635     }%
4636     {%
4637         \GlossariesWarning{Empty glossary for
4638             \string\printnoidxglossary[type={##1}].
4639             Rerun may be required (or you may have forgotten to use
4640             commands like \string\gls)}%
4641     }%
4642 }%

```

Adjust display number list to check for type:

```

4643     \renewcommand*\@glsdisplaynumberlist}[1]{%
4644         \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
4645         {\@glsxtr@idx@displaynumberlist{##1}}%
4646         {\@glsxtr@noidx@displaynumberlist{##1}}%
4647     }%

```

Adjust entry list:

```

4648     \renewcommand*\@glsentrynumberlist}[1]{%
4649         \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
4650         {\@glsxtr@idx@entrynumberlist{##1}}%
4651         {\@glsxtr@noidx@entrynumberlist{##1}}%
4652     }%

```

Adjust number list loop

```

4653     \renewcommand*\@glsnumberlistloop}[2]{%
4654         \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
4655     {%
4656         \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
4657             not available for glossary '##1'}{%
4658         }%
4659         {\@glsxtr@noidx@numberlistloop{##1}{##2}}%
4660     }%

```

Only sanitize sort for normal indexing glossaries.

```

4661     \renewcommand*\@glsprestandardsort}[3]{%
4662         \expandafter\DTLifinlist\expandafter{##2}{\@glsxtr@reg@glosslist}%
4663     {%
4664         \glsdosanitizesort
4665     }%
4666     {%
4667         \ifglssanitizesort
4668             \@gls@noidx@sanitizesort
4669         \else
4670             \@gls@noidx@nosanitizesort

```

```

4671         \fi
4672     }%
4673 }%

```

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

```

4674     \renewcommand*{\new@glossaryentry[2]}{%
4675         \PackageError{glossaries-extra}{Glossary entries must be defined
4676             in the preamble\MessageBreak when you use the optional argument
4677             of \string\makeglossaries}{Either move your definitions to the
4678             preamble or don't use the optional argument of
4679             \string\makeglossaries}%
4680     }%

```

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).

```

4681     \let\@glo@assign@sortkey\@glstr@mixed@assign@sortkey
4682     \renewcommand*{\@printgloss@setsort}{%

```

Need to extract just the type value.

```

4683         \expandafter\@glstr@gettype\expandafter,\@glstr@printglossopts,%
4684         type=\glsdefaulttype,\@end@glstr@gettype
4685         \def\@glo@sorttype{\@glo@default@sorttype}%
4686     }%

```

Check automake setting:

```

4687     \ifglautomake
4688     \renewcommand*{\@gls@doautomake}{%
4689         \for\@gls@type:=\@glstr@reg@glosslist\do{%
4690             \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
4691         }%
4692     }%
4693 \fi

```

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

```

4694     \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
4695     \fi
4696 }%
4697 }%
4698 }%
4699 }

```

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

Earlier versions of `glossaries-extra` simply saved the original version of `\@printglossary` with `\let \@glstr@orgprintglossary`. This was later changed to actually defining `\@glstr@orgprintglossary` to something similar with some alterations to allow for ignored glossaries, which don't have an associated title and to by-pass the existence check with `\ifglossaryexists` which doesn't recognise ignored glossaries. (bib2gls writes `\provideignoredglossary` to the `gls.tex` file for some settings, so the glossary might not been defined on the first \LaTeX run and it needs to be allowed with `\printunsrtglossary` on subsequent runs.)

Unfortunately, removing the existence check will cause an error if `\printglossary` is used with an ignored glossary.

As from glossaries v4.46, some new commands have been included to allow the existence check to be varied depending on whether or not ignored glossaries should be allowed, so check for them:

oss@checkexists

```
4700 \ifdef\@printgloss@checkexists
4701 {\newcommand{\glxtr@printgloss@checkexists}{\@printgloss@checkexists}}
4702 {\newcommand{\glxtr@printgloss@checkexists}[2]{#2}}
```

rgprintglossary (This command is also used for on-the-fly setting.)

```
4703 \newcommand{\@glxtr@orgprintglossary}[2]{%
4704   \def\@glo@type{\gldefaulttype}%
```

Add check here.

```
4705   \def\glossarytitle{%
4706     \ifcsdef{\@glo@type@\@glo@type @title}%
4707       {\csuse{\@glo@type@\@glo@type @title}}%
4708       {\glossaryname}}%
4709   \def\glossarytoctitle{\glossarytitle}%
4710   \let\org@glossarytitle\glossarytitle
4711   \def\@glossarystyle{%
4712     \ifx\@glossary@default@style\relax
4713       \GlossariesWarning{No default glossary style provided \MessageBreak
4714         for the glossary '\@glo@type'. \MessageBreak
4715         Using deprecated fallback. \MessageBreak
4716         To fix this set the style with \MessageBreak
4717         \string\setglossarystyle\space or use the \MessageBreak
4718         style key=value option}%
4719     \fi
4720   }%
4721   \def\gls@dotoc@title{\gls@settoc@title{\@glo@type}}%
4722   \let\org@glossaryentrynumbers\glossaryentrynumbers
4723   \bgroup
4724     \@printgloss@setsort
4725     \setkeys{printgloss}{#1}%
4726     \ifx\glossarytitle\org@glossarytitle
4727     \else
4728       \cslet{\@glo@type@\@glo@type @title}{\glossarytitle}%
4729     \fi
4730     \let\currentglossary\@glo@type
4731     \let\org@glossaryentrynumbers\glossaryentrynumbers
4732     \let\glsnonextpages\@glsnonextpages
4733     \let\glsnextpages\@glsnextpages
4734     \glxtractivatenopost
4735     \gls@dotoc@title
4736     \@glossarystyle
```



```

4737 \let\gls@org@glossaryentryfield\glossentry
4738 \let\gls@org@glossarysubentryfield\subglossentry
4739 \renewcommand{\glossentry}[1]{%
4740   \protected@xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
4741   \gls@org@glossaryentryfield{##1}%
4742 }%
4743 \renewcommand{\subglossentry}[2]{%
4744   \protected@xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4745   \gls@org@glossarysubentryfield{##1}{##2}%
4746 }%
4747 \@gls@preglossaryhook
4748 \glstr@printgloss@checkexists{\@glo@type}{#2}%
4749 \egroup
4750 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
4751 \global\let\warn@noprntglossary\relax
4752 }

```

ractivatenopost Change \nopostdesc and \glstrnopostpunc to behave as they do in the glossary.

```

4753 \newcommand*{\glstractivatenopost}{%
4754   \let\nopostdesc\@nopostdesc
4755   \let\glstrnopostpunc\@glstr@nopostpunc
4756 }

```

lsxtrnopostpunc

```

4757 \newrobustcmd*{\glstrnopostpunc}{%

```

sxtr@nopostpunc Provide a command that works like \nopostdesc but only switches of the punctuation without suppressing the post-description hook.

```

4758 \newcommand{\@glstr@nopostpunc}{%
4759   \let\@@glstr@org@postdescription\glspostdescription
4760   \ifglsnopostdot
4761     \renewcommand{\glspostdescription}{%
4762       \glsnopostdottrue
4763       \let\glspostdescription\@@glstr@org@postdescription
4764       \let\glstrrestorepostpunc\@glstr@restore@postpunc
4765       \glstrpostdescription
4766       \@glstr@nopostpunc@postdesc}%
4767   \else
4768     \renewcommand{\glspostdescription}{%
4769       \let\glspostdescription\@@glstr@org@postdescription
4770       \let\glstrrestorepostpunc\@glstr@restore@postpunc
4771       \glstrpostdescription
4772       \@glstr@nopostpunc@postdesc}%
4773   \fi
4774   \glsnopostdotfalse
4775 }

```

stpunc@postdesc

```

4776 \newcommand*{\@glstr@nopostpunc@postdesc}{%

```

restore@postpunc

```
4777 \newcommand*{\@glxtr@restore@postpunc}{%
4778 \def\@glxtr@nopostpunc@postdesc{%
4779 \@glxtr@org@postdescription
4780 \let\@glxtr@nopostpunc@postdesc\@empty
4781 \let\glxtrrestorepostpunc\@empty
4782 }%
4783 }
```

restorepostpunc Does nothing outside of glossary.

```
4784 \newcommand*{\glxtrrestorepostpunc}{}%
```

\@printglossary Redefine.

```
4785 \renewcommand{\@printglossary}[2]{%
4786 \def\@glxtr@printglossopts{#1}%
4787 \@glxtr@orgprintglossary{#1}{#2}%
4788 }
```

Add a key that switches off the entry targets:

```
4789 \define@choicekey{printgloss}{target}
4790 [\@glxtr@printglossval\@glxtr@printglossnr]%
4791 {true,false}[true]%
4792 {%
4793 \ifcase\@glxtr@printglossnr
4794 \def\@glstarget{\@glsdohypertarget}%
4795 \else
4796 \let\@glstarget\@secondoftwo
4797 \fi
4798 }
```

hypernameprefix

```
4799 \newcommand{\@glxtrhypernameprefix}{}%
```

New to v1.20:

```
4800 \define@key{printgloss}{targetnameprefix}{%
4801 \renewcommand{\@glxtrhypernameprefix}{#1}%
4802 }

4803 \define@key{printgloss}{prefix}{%
4804 \renewcommand{\@glolinkprefix}{#1}%
4805 }

4806 \define@key{printgloss}{label}{%
4807 \glxtrsetglossarylabel{#1}%
4808 }
```

etglossarylabel Set the label for subsequent glossaries. If the label is fixed (that is, doesn't change with each glossary) this will need to be scoped or changed again to prevent duplicate labels.

```
4809 \newcommand{\glxtrsetglossarylabel}[1]{%
4810   \renewcommand*{\@glossaryseclabel}{%
4811     \protected@edef\@currentlabelname{\glossarytoctitle}%
4812     \label{#1}%
4813   }%
4814 }
```

xtr@leveloffset

```
4815 \newcount\@glxtr@leveloffset
```

New to v1.44:

```
4816 \define@key{printgloss}{leveloffset}{%
4817   \@glxtr@assign@leveloffset#1\relax
4818 }
```

ign@leveloffset

```
4819 \newcommand*{\@glxtr@assign@leveloffset}{%
4820   \@ifnextchar+{\p@glxtr@assign@leveloffset}{\np@glxtr@assign@leveloffset}%
4821 }
```

ign@leveloffset Discard initial + character.

```
4822 \newcommand*{\p@glxtr@assign@leveloffset}[1]{%
4823   \@ifnextchar+{\pp@glxtr@assign@leveloffset}{\np@glxtr@assign@leveloffset}%
4824 }
```

ign@leveloffset

```
4825 \def\np@glxtr@assign@leveloffset#1\relax{\@glxtr@leveloffset=#1\relax}
```

ign@leveloffset

```
4826 \def\pp@glxtr@assign@leveloffset#1\relax{\advance\@glxtr@leveloffset by #1\relax}
```

```
4827 \define@boolkey{printgloss}[glxtr@printgloss@]{groups}[true]{}
4828 \glxtr@printgloss@groupstrue
```

lsdohypertarget Redefine to insert \@glxtrhypernameprefix before the target name.

```
4829 \let\@glxtr@org@glsdohypertarget\glsdohypertarget
4830 \renewcommand{\glsdohypertarget}[2]{%
4831   \@glxtr@org@glsdohypertarget{\@glxtrhypernameprefix#1}{#2}%
4832 }
```

Update \@glstarget to use \def instead being assigned with \let so that it can pick up the new definition and allow any further redefinitions:

```
4833 \ifx\@glstarget\@glxtr@org@glsdohypertarget
4834   \def\@glstarget{\glsdohypertarget}%
4835 \fi
```

r@do@org@target Provide a way to locally do the original.

```
4836 \newcommand{\@glxtr@do@org@target}[2]{%
4837   {%
4838     \let\glsdohypertarget\glxtr@org@glsdohypertarget
4839     \@glstarget{#1}{#2}%
4840   }%
4841 }
```

@makeglossaries For the benefit of makeglossaries

```
4842 \newcommand*{\glxtr@makeglossaries}[1]{}
```

@glxtr@gettype Get just the type.

```
4843 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
4844   \def\@glo@type{#2}%
4845 }
```

@assign@sortkey Assign the sort key.

```
4846 \newcommand\@glxtr@mixed@assign@sortkey[1]{%

4847   \protected@edef\@glo@type{\@glo@type}%
4848   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
4849   {%
4850     \@glo@no@assign@sortkey{#1}%
4851   }%
4852   {%
4853     \@glo@assign@sortkey{#1}%
4854   }%
4855 }%
```

Display number list for the regular version:

splaynumberlist

```
4856 \let\@glxtr@idx@displaynumberlist\glstdisplaynumberlist
```

Display number list for the “noidx” version:

splaynumberlist

```
4857 \newcommand*{\@glxtr@noidx@displaynumberlist}[1]{%
4858   \letcs{\@gls@loclist}{glo@\glstdetoklabel{#1}@loclist}%
4859   \ifdef\@gls@loclist
4860   {%
4861     \def\@gls@noidxloclist@sep{%
4862       \def\@gls@noidxloclist@sep{%
4863         \def\@gls@noidxloclist@sep{%
4864           \glsnumlistsep
4865         }%
4866         \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
4867       }%
4868     }%
```

```

4869 \def\@gls@noidxloclist@finalsep{}%
4870 \def\@gls@noidxloclist@prev{}%
4871 \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
4872 \@gls@noidxloclist@finalsep
4873 \@gls@noidxloclist@prev
4874 }%
4875 {%

4876 \glstrundef\tag
4877 \glsdoifexists{#1}%
4878 {%
4879 \GlossariesWarning{Missing location list for ‘#1’. Either
4880 a rerun is required or you haven’t referenced the entry.}%
4881 }%
4882 }%
4883 }%
4884

```

And for the number list loop:

@numberlistloop

```

4885 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
4886 \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
4887 \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
4888 \let\@gls@org@glsseeformat\glsseeformat
4889 \let\glsnoidxdisplayloc#2\relax
4890 \let\glsseeformat#3\relax
4891 \ifdef\@gls@loclist
4892 {%
4893 \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
4894 }%
4895 {%

4896 \glstrundef\tag
4897 \glsdoifexists{#1}%
4898 {%
4899 \GlossariesWarning{Missing location list for ‘##1’. Either
4900 a rerun is required or you haven’t referenced the entry.}%
4901 }%
4902 }%
4903 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
4904 \let\glsseeformat\@gls@org@glsseeformat
4905 }%

```

Same for entry number list.

entrynumberlist

```

4906 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
4907 \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
4908 \ifdef\@gls@loclist
4909 {%

```

```

4910 \glsnoidxloclist{\@gls@loclist}%
4911 }%
4912 {%

4913 \glsxtrundeftag
4914 \glsdoifexists{#1}%
4915 {%
4916 \GlossariesWarning{Missing location list for ‘#1’. Either
4917 a rerun is required or you haven’t referenced the entry.}%
4918 }%
4919 }%
4920 }%

```

entrynumberlist

```

4921 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

4922 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
4923 \protected@edef\@glsxtr@titlelabel{#1}%
4924 \ifdefvoid\@glsxtr@titlelabel
4925 {}%
4926 {%
4927 \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle@#1}}%
4928 }%
4929 \ifdefvoid{\@glsxtr@titlelabel}%
4930 {%
4931 \DTLifint{#1}%
4932 {%
4933 \ifnum#1<256\relax
4934 \edef#2{\char#1\relax}%
4935 \else
4936 \edef#2{#1}%
4937 \fi
4938 }%
4939 {%
4940 \ifcsundef{#1groupname}%
4941 {\def#2{#1}}%
4942 {\letcs#2{#1groupname}}%
4943 }%
4944 }%
4945 {%
4946 \let#2\@glsxtr@titlelabel
4947 }%
4948 }

```

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

```

4949 \let\glsxtr@org@getgrouptitle\@gls@getgrouptitle

```

`trgetgrouptitle` Provide a user-level command to fetch the group title. The first argument is the group label. The second argument is a control sequence in which to store the title.

```
4950 \newrobustcmd{\glxtrgetgrouptitle}[2]{%
4951   \protected@edef\@glxtr@titlelabel{glxtr@grouptitle@#1}%
4952   \@onelevel@sanitize\@glxtr@titlelabel
4953   \ifcsdef{\@glxtr@titlelabel}
4954     {\letcs{#2}{\@glxtr@titlelabel}}%
4955     {\glxtr@org@getgrouptitle{#1}{#2}}%
4956 }
4957 \let\@gls@getgrouptitle\glxtrgetgrouptitle
```

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

```
4958 \newcommand{\glxtrsetgrouptitle}[2]{%
4959   \protected@edef\@glxtr@titlelabel{glxtr@grouptitle@#1}%
4960   \@onelevel@sanitize\@glxtr@titlelabel
4961   \protected@csxdef{\@glxtr@titlelabel}{#2}%
4962 }
```

`alsetgrouptitle` As above put only locally defines the title.

```
4963 \newcommand{\glxtrlocalsetgrouptitle}[2]{%
4964   \protected@edef\@glxtr@titlelabel{glxtr@grouptitle@#1}%
4965   \@onelevel@sanitize\@glxtr@titlelabel
4966   \protected@csedef{\@glxtr@titlelabel}{#2}%
4967 }
```

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

```
4968 \renewcommand*{\glsnavigation}{%
4969   \def\@gls@between{}%
4970   \ifcsundef{@gls@hypergroup@list@\@glo@type}%
4971     {%
4972       \def\@gls@list{}%
4973     }%
4974     {%
4975       \expandafter\let\expandafter\@gls@list
4976         \csname @gls@hypergroup@list@\@glo@type\endcsname
4977     }%
4978     \@for\@gls@tmp:=\@gls@list\do{%
4979       \@gls@between
4980       \glxtrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
4981       \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
4982       \let\@gls@between\glshypernavsep
4983     }%
4984 }
```

`@noidx@glossary`

```
4985 \renewcommand*{\@print@noidx@glossary}{%
4986   \ifcsdef{@gls@ref@\@glo@type}%
4987     {%
```

```

4988 \ifcsdef{@glo@sortmacro@\@glo@sorttype}%
4989 {%
4990 \csuse{@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
4991 }%
4992 {%
4993 \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
4994 }%
4995 \glossarysection[\glossarytoctitle]{\glossarytitle}%
4996 \glossarypreamble

```

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

```

4997 \def\@gls@currentlettergroup{%
4998 \begin{theglossary}%
4999 \glossaryheader
5000 \glsresetentrylist
5001 \forlistcsloop{\@gls@noidx@do}{\@glsref@\@glo@type}%
5002 \end{theglossary}%
5003 \glossarypostamble
5004 }%
5005 {%

```

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

```

5006 \glsxtrifemptyglossary{\@glo@type}%
5007 {}%
5008 {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
5009 \@gls@noref@warn{\@glo@type}%
5010 }%
5011 }

```

noidxdisplayloc Patch to check for range formations.

```

5012 \renewcommand*{\glsnoidxdisplayloc}[4]{%
5013 \setentrycounter[#1]{#2}%
5014 \@glsxtr@display@loc#3\empty\end@glxtr@display@loc{#4}%
5015 }

```

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

```

5016 \def\@glsxtr@display@loc#1#2\end@glxtr@display@loc#3{%
5017 \ifx#1\relax
5018 \glsxtrdisplaystartloc{#2}{#3}%
5019 \else
5020 \ifx#1\relax
5021 \glsxtrdisplayendloc{#2}{#3}%
5022 \else
5023 \glsxtrdisplaysingleloc{#1#2}{#3}%
5024 \fi
5025 \fi
5026 }

```


`\displaysingleloc` Single location.

```
5027 \newcommand*{\glxtrdisplaysingleloc}[2]{%
5028   \csuse{#1}{#2}%
5029 }
```

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

`\displaystartloc` Start of a location range.

```
5030 \newcommand*{\glxtrdisplaystartloc}[2]{%
5031   \protected@edef\glxtrlocrangefmt{#1}%
5032   \ifx\glxtrlocrangefmt\empty
5033     \def\glxtrlocrangefmt{\glsnnumberformat}%
5034   \fi
5035   \expandafter\glxtrdisplaysingleloc
5036   \expandafter{\glxtrlocrangefmt}{#2}%
5037 }
```

`\trdisplayendloc` End of a location range.

```
5038 \newcommand*{\glxtrdisplayendloc}[2]{%
5039   \protected@edef\@glxtr@tmp{#1}%
5040   \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{\glsnnumberformat}}{}%
5041   \ifx\glxtrlocrangefmt\@glxtr@tmp
5042     \else
5043       \GlossariesExtraWarning{Mismatched end location range
5044         (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
5045     \fi
5046     \expandafter\glxtrdisplayendloohook\expandafter{\@glxtr@tmp}{#2}%
5047     \expandafter\glxtrdisplaysingleloc
5048     \expandafter{\glxtrlocrangefmt}{#2}%
5049   \def\glxtrlocrangefmt{}%
5050 }
```

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

```
5051 \newcommand*{\glxtrdisplayendloohook}[2]{}
```

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

```
5052 \newcommand*{\glxtrlocrangefmt}{}%
```

`\setentrycounter` Adjust `\setentrycounter` to save the original prefix.

```
5053 \renewcommand*{\setentrycounter}[2][{}]{%
5054   \def\glxtrcounterprefix{#1}%
5055   \ifx\glxtrcounterprefix\@empty
5056     \def\@glo@counterprefix{.}%
5057   \else
5058     \def\@glo@counterprefix{.#1.}%
5059   \fi
5060   \def\glsetentrycounter{#2}%
5061 }
```

ls@removespaces Redefine to allow adjustments to location hyperlink.

```
5062 \def\@gls@removespaces#1 #2\@nil{%
5063   \toks@=\expandafter{\the\toks@#1}%
5064   \ifx\\#2\\%
```

```
5065     \edef\@glo@tmp{\the\toks@}%
5066     \ifx\@glo@tmp\empty
5067     \else
```

Expand location (just in case \toks@ is needed for something else).

```
5068     \expandafter\glsxtrlocationhyperlink\expandafter
5069     \glstentrycounter\expandafter\@glo@counterprefix\expandafter{\the\toks@}%
5070     \fi
5071 \else
5072   \@gls@ReturnAfterFi{%
5073     \@gls@removespaces#2\@nil
5074   }%
5075 \fi
5076 }
```

cationhyperlink

`\glsxtrlocationhyperlink{<counter>}{<prefix>}{<location>}`

```
5077 \newcommand*\glsxtrlocationhyperlink}[3]{%
5078   \ifdefvoid\glsxtrsupplocationurl
5079   {%
5080     \GlsXtrInternalLocationHyperlink{#1}{#2}{#3}%
5081   }%
5082   {%
5083     \hyperref{\glsxtrsupplocationurl}{#1#2#3}{#3}%
5084   }%
5085 }
```

supphypernumber

```
5086 \newcommand*\glsxtrsupphypernumber}[1]{%
5087   {%
5088     \glshasattribute{\glscurrententrylabel}{externallocation}%
5089     {%
5090       \def\glsxtrsupplocationurl{%
5091         \glsgetattribute{\glscurrententrylabel}{externallocation}}%
5092       }%
5093     {%
5094       \def\glsxtrsupplocationurl{}%
5095     }%
5096     \glshypernumber{#1}%
5097   }%
5098 }
```

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

@print@glossary

```

5099 \renewcommand{\@print@glossary}{%
5100   \makeatletter
5101   \@input@{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
5102   \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
5103   {}%
5104   {\glstrNoGlossaryWarning{\@glo@type}}%
5105   \ifglxindy
5106     \ifcsundef{@xdy@\@glo@type @language}%
5107     {%
5108       \edef\@do@auxoutstuff{%
5109         \noexpand\AtEndDocument{%
5110           \noexpand\immediate\noexpand\write\@auxout{%
5111             \string\providecommand\string\@xdylanguage[2]{}}%
5112           \noexpand\immediate\noexpand\write\@auxout{%
5113             \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
5114         }%
5115       }%
5116     }%
5117     {%
5118       \edef\@do@auxoutstuff{%
5119         \noexpand\AtEndDocument{%
5120           \noexpand\immediate\noexpand\write\@auxout{%
5121             \string\providecommand\string\@xdylanguage[2]{}}%
5122           \noexpand\immediate\noexpand\write\@auxout{%
5123             \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
5124               @language\endcsname}}%
5125         }%
5126       }%
5127     }%
5128     \@do@auxoutstuff
5129     \edef\@do@auxoutstuff{%
5130       \noexpand\AtEndDocument{%
5131         \noexpand\immediate\noexpand\write\@auxout{%
5132           \string\providecommand\string\@gls@codepage[2]{}}%
5133         \noexpand\immediate\noexpand\write\@auxout{%
5134           \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
5135       }%
5136     }%
5137     \@do@auxoutstuff
5138   \fi
5139   \renewcommand*{\@warn@nomakeglossaries}{%
5140     \GlossariesWarningNoLine{\string\makeglossaries\space
5141       hasn't been used,^^Jthe glossaries will not be updated}%
5142   }%
5143 }
```

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

`oGlsWarningHead` Header message.

```
5144 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
5145   This document is incomplete. The external file associated with
5146   the glossary ‘#1’ (which should be called \texttt{#2})
5147   hasn’t been created.%
5148 }
```

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

```
5149 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
5150   This has probably happened because there are no entries defined
5151   in this glossary.%
5152 }
```

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

```
5153 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
5154   If you don’t want this glossary,
5155   add \texttt{nomain} to your package option list when you load
5156   \texttt{glossaries-extra.sty}. For example:%
5157 }
```

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

```
5158 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
5159   Did you forget to use \texttt{type=#1} when you defined your
5160   entries? If you tried to load entries into this glossary with
5161   \texttt{\string\loadglsentries} did you remember to use
5162   \texttt{[#1]} as the optional argument? If you did, check that
5163   the definitions in the file you loaded all had the type set
5164   to \texttt{\string\glsdefaulttype}.%
5165 }
```

`arningCheckFile` Advisory message to check the file contents.

```
5166 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
5167   Check the contents of the file \texttt{#1}. If
5168   it’s empty, that means you haven’t indexed any of your entries in this
5169   glossary (using commands like \texttt{\string\gls} or
5170   \texttt{\string\glsadd}) so this list can’t be generated.
5171   If the file isn’t empty, the document build process hasn’t been
5172   completed.%
5173 }
```

`WarningAutoMake` Message when automake option has been used.

```
5174 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
5175   You may need to rerun \LaTeX. If you already have, it may be that
5176   \TeX’s shell escape doesn’t allow you to run
5177   \ifglxindy xindy\else makeindex\fi. Check the
5178   transcript file \texttt{\jobname.log}. If the shell escape is
```

```

5179 disabled, try one of the following:
5180
5181 \begin{itemize}
5182   \item Run the external (Lua) application:
5183
5184     \texttt{makeglossaries-lite \string"\jobname\string"}
5185
5186   \item Run the external (Perl) application:
5187
5188     \texttt{makeglossaries \string"\jobname\string"}
5189 \end{itemize}
5190
5191 Then rerun \LaTeX\ on this document.
5192 \GlossariesExtraWarning{Rerun required to build the
5193 glossary ‘#1’ or check TeX’s shell escape allows
5194 you to run \ifglxindy xindy\else makeindex\fi}%
5195 }

```

WarningMisMatch Mismatching \makenoidxglossaries.

```

5196 \newcommand{\GlsXtrNoGlsWarningMisMatch}{%
5197   You need to either replace \texttt{\string\makenoidxglossaries}
5198   with \texttt{\string\makeglossaries} or replace
5199   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
5200   \texttt{\string\printnoidxglossary}
5201   (or \texttt{\string\printnoidxglossaries}) and then rebuild
5202   this document.%
5203 }

```

arningBuildInfo Build advice.

```

5204 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
5205   Try one of the following:
5206   \begin{itemize}
5207     \item Add \texttt{automake} to your package option list when you load
5208       \texttt{glossaries-extra.sty}. For example:
5209
5210         \texttt{\string\usepackage[automake]%
5211           \glsgopenbrace glossaries-extra\glsgclosebrace}
5212
5213     \item Run the external (Lua) application:
5214
5215       \texttt{makeglossaries-lite.lua \string"\jobname\string"}
5216
5217     \item Run the external (Perl) application:
5218
5219       \texttt{makeglossaries \string"\jobname\string"}
5220   \end{itemize}
5221
5222   Then rerun \LaTeX\ on this document.%
5223 }

```

trRecordWarning Paragraph for record=only.

```
5224 \newcommand{\GlsXtrRecordWarning}[1]{%
5225 \texttt{\string\printglossary} doesn't work
5226 with the \texttt{record=@glxtr@record@setting} package option
5227 use\par\texttt{\string\printunsrtglossary[type=#1]}\par
5228 instead (or change the package option).%
5229 }
```

oGlsWarningTail Final paragraph.

```
5230 \newcommand{\GlsXtrNoGlsWarningTail}{%
5231 This message will be removed once the problem has been fixed.%
5232 }
```

GlsWarningNoOut No out file created. Build advice.

```
5233 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
5234 The file \texttt{#1} doesn't exist. This most likely means you haven't used
5235 \texttt{\string\makeglossaries} or you have used
5236 \texttt{\string\nofiles}. If this is just a draft version of the
5237 document, you can suppress this message using the
5238 \texttt{nomissingglstext} package option.%
5239 }
```

glossarywarning

```
5240 \newcommand*{\@glxtr@defaultnoglossarywarning}[1]{%
5241 \glossarysection[\glossarytoctitle]{\glossarytitle}
5242 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
5243 \par
5244 \glxtrifemptyglossary{#1}%
5245 {%
5246 \GlsXtrNoGlsWarningEmptyStart\space
5247 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
5248 \medskip
5249 \noindent\texttt{\string\usepackage[nomain\ifglssacronym ,acronym\fi]%
5250 \glsopenbrace glossaries-extra\glsclosebrace}
5251 \medskip
5252 }%
5253 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
5254 }%
5255 {%
5256 \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
5257 {%
5258 \GlsXtrNoGlsWarningCheckFile
5259 {\jobname.\csname @glotype@\@glo@type @out\endcsname}
5260
5261 \ifglssautomake
5262
5263 \GlsXtrNoGlsWarningAutoMake{#1}
5264
5265 \else
```

```

5266
5267     \ifthenelse{\equal{#1}{main}}{%
5268     {%
5269         \GlsXtrNoGlsWarningEmptyMain\par
5270         \medskip
5271         \noindent\texttt{\string\usepackage[nomain]%
5272             \glsopenbrace glossaries-extra\glsclosebrace}
5273         \medskip
5274     }%
5275     }%
5276
5277     \ifdefequal\makeglossaries\@no@makeglossaries
5278     {%
5279         \GlsXtrNoGlsWarningMisMatch
5280     }%
5281     {%
5282         \GlsXtrNoGlsWarningBuildInfo
5283     }%
5284     \fi
5285 }%
5286 {%
5287     \GlsXtrNoGlsWarningNoOut
5288     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
5289 }%
5290 }%
5291 \par
5292 \GlsXtrNoGlsWarningTail
5293 }

```

glossarywarning Warn about using `\printglossary` with `record`

```

5294 \newcommand*{\@glxtr@record@noglossarywarning}[1]{%
5295     \GlossariesExtraWarning{\string\printglossary\space doesn't work\MessageBreak
5296     with record=\@glxtr@record@setting\space package option\MessageBreak(use
5297     \string\printunsrtglossary[type=#1])\MessageBreak
5298     instead (or change the package option)}%
5299     \glossarysection[\glossarytoctitle]{\glossarytitle}
5300     \GlsXtrRecordWarning{#1}
5301     \GlsXtrNoGlsWarningTail
5302 }

```

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

ResourceOptions Default resource options.

```

5303 \newcommand*{\GlsXtrDefaultResourceOptions}{}

```

xtresourcefile 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.

```

5304 \newcommand*{\glxtrresourcefile}[2][ ]{%

```

The record option can't be set after this command.

```

5305 \disable@keys{glossaries-extra.sty}{record}%
5306 \glstr@writefields
5307 \ifdefempty\GlsXtrDefaultResourceOptions
5308 {%
5309   \protected@write\@auxout{\glstrresourceinit}%
5310   {\string\glstr@resource{#1}{#2}}%
5311 }%
5312 {%
5313   \protected@write\@auxout{\glstrresourceinit}%
5314   {\string\glstr@resource{\GlsXtrDefaultResourceOptions,#1}{#2}}%
5315 }%
5316 \let\@glstr@org@see@noindex\@glstr@see@noindex
5317 \let\@glstr@see@noindex\relax
5318 \IfFileExists{#2.glstex}%
5319 {%

```

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

```

5320 \edef\@bibglstr@restreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
5321 \makeatletter
5322 \@input{#2.glstex}%
5323 \@bibglstr@restreat

```

If the record=nameref option has been set, check if this is supported by the installed version of bib2gls.

```

5324 \glstr@check@bibglstr@nameref
5325 }%
5326 {%
5327   \GlossariesExtraWarning{No file '#2.glstex'}%
5328 }%
5329 \let\@glstr@see@noindex\@glstr@org@see@noindex
5330 }
5331 \@onlypreamble\glstrresourcefile

```

@bibglstr@nameref This will only warn after bib2gls has created the .glstex file, but there's way to check before.

```

5332 \newcommand{\@glstr@check@bibglstr@nameref}{%
5333   \ifx\@glstr@record@setting\@glstr@record@setting@nameref
5334     \ifdef\bibglshrefchar
5335       {}%
5336     {%
5337       \GlossariesExtraWarning{record=nameref requires at least
5338       version 1.8 of bib2gls}%
5339     }%
5340   \fi
5341   \let\@glstr@check@bibglstr@nameref\relax
5342 }

```

glstrresourceinit Code used during the protected write operation.

```

5343 \newcommand*{\glstrresourceinit}{}

```


trresourcecount

```
5344 \newcount\glxtrresourcecount
```

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

```
5345 \newcommand*{\GlsXtrLoadResources}[1][{}]{%
5346   \ifnum\glxtrresourcecount=0\relax
5347     \glxtrresourcefile[#1]{\jobname}%
5348   \else
5349     \glxtrresourcefile[#1]{\jobname-\the\glxtrresourcecount}%
5350   \fi
5351   \advance\glxtrresourcecount by 1\relax
5352 }
```

glxtr@resource

```
5353 \newcommand*{\glxtr@resource}[2]{}
```

\glxtr@fields

```
5354 \newcommand*{\glxtr@fields}[1]{}
```

xtr@texencoding

```
5355 \newcommand*{\glxtr@texencoding}[1]{}
```

\glxtr@langtag

```
5356 \newcommand*{\glxtr@langtag}[1]{}
```

@pluralsuffixes

```
5357 \newcommand*{\glxtr@pluralsuffixes}[4]{}
```

tr@shortcutsval

```
5358 \newcommand*{\glxtr@shortcutsval}[1]{}
```

sxtr@linkprefix

```
5359 \newcommand*{\glxtr@linkprefix}[1]{}
```

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

```
5360 \newcommand*{\glxtr@writefields}{%

5361   \protected@write\@auxout{%
5362     {\string\providecommand*{\string\glxtr@fields}[1]{}}%
5363   \protected@write\@auxout{%
5364     {\string\providecommand*{\string\glxtr@resource}[2]{}}%
5365   \protected@write\@auxout{%
5366     {\string\providecommand*{\string\glxtr@pluralsuffixes}[4]{}}%
5367   \protected@write\@auxout{%
5368     {\string\providecommand*{\string\glxtr@shortcutsval}[1]{}}%
5369   \protected@write\@auxout{%
5370     {\string\providecommand*{\string\glxtr@linkprefix}[1]{}}%
5371   \protected@write\@auxout{{\string\glxtr@fields{\@gls@keymap}}}%

```

```

5372 \protected@write\@auxout{}%
5373 {\string\providecommand*\string\glxtr@record}[5]{}%

5374 \ifx\@glxtr@record@setting\@glxtr@record@setting@nameref
5375 \protected@write\@auxout{}%
5376 {\string\providecommand*\string\glxtr@record@nameref}[8]{}%
5377 \fi

```

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

```

5378 \ifdef\CurrentTrackedLanguageTag
5379 {%
5380 \protected@write\@auxout{}%
5381 \string\glxtr@langtag{\CurrentTrackedLanguageTag}%
5382 }%
5383 {}%
5384 \protected@write\@auxout{}\string\glxtr@pluralsuffixes
5385 {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
5386 {\glxtrabbrvpluralsuffix}%
5387 \ifdef\inputencodingname
5388 {%
5389 \protected@write\@auxout{}\string\glxtr@texencoding{\inputencodingname}%
5390 }%
5391 {%

```

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.)

```

5392 \ifpackageloaded{fontspec}%
5393 {\protected@write\@auxout{}\string\glxtr@texencoding{utf8}}%
5394 {}%
5395 }%
5396 \protected@write\@auxout{}\string\glxtr@shortcutsval{\@glxtr@shortcutsval}%

```

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

```

5397 \AtBeginDocument
5398 {\protected@write\@auxout{}\string\glxtr@linkprefix{\glolinkprefix}}%
5399 \let\glxtr@writefields\relax

```

If the `automake` option is on, try running `bib2gls` if the aux file exists. This has to be done before the aux file is opened (so package options `automake=immediate` and `automake=true` are identical if just `bib2gls` is used). The double-quotes around `\jobname` have been removed (v1.19) since `\jobname` will include double-quotes if the file name has spaces.

```

5400 \ifglautomake
5401 \IfFileExists{\jobname.aux}%
5402 {\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`.

```

5403 \ifx\@gls@doautomake\@gls@doautomake@err
5404 \let\@gls@doautomake\relax
5405 \fi
5406 \fi

```

Check if order=letter has been used by mistake (but not if record=alsoindex has been used).

```

5407 \@glsxtr@if@record@only
5408 {\ifdefstring{\glsorder}{letter}%
5409 {\GlossariesExtraWarningNoLine{Package option ‘order=letter’ isn’t
5410 supported with ‘record=\@glsxtr@record@setting’. Use ‘break-at=none’
5411 resource option instead}}}%
5412 {}%
5413 }%
5414 {}%
5415 }

```

do@automake@err

```

5416 \newcommand*{\@gls@doautomake@err}{%
5417 \PackageError{glossaries}{You must use
5418 \string\makeglossaries\space with automake=true}
5419 {%
5420 Either remove the automake=true setting or
5421 add \string\makeglossaries\space to your document preamble.%
5422 }%
5423 }

```

Allow locations specific to a particular counter to be recorded.

\glsxtr@record

```

5424 \newcommand*{\glsxtr@record}[5]{}

```

@record@nameref Used with record=nameref to include current label information.

```

5425 \newcommand*{\glsxtr@record@nameref}[8]{}

```

r@counterrecord Aux file command.

```

5426 \newcommand*{\glsxtr@counterrecord}[3]{%
5427 \glsxtrfieldlistgadd{#1}{record.#2}{#3}%
5428 }

```

unterrecordhook Hook used by \@glsxtr@dorecord.

```

5429 \newcommand*{\@glsxtr@counterrecordhook}{}

```

trRecordCounter Activate recording for a particular counter (identified in the argument).

```

5430 \newcommand*{\GlsXtrRecordCounter}[1]{%
5431 \@glsxtr@recordcounter{#1}%
5432 }
5433 \@onlypreamble\GlsXtrRecordCounter

```

docounterrecord

```

5434 \newcommand*{\@glxtr@docounterrecord}[1]{%
5435   \protected@write\@auxout{}\string\glxtr@counterrecord
5436     {\@glslabel}{#1}{\csuse{the#1}}}%
5437 }

```

glxtrglossentry

Users may prefer to have entries displayed throughout the document rather than gathered together in a list. This command emulates the way `\glossentry` behaves (without the style formatting commands like `\item`). This needs to define `\currentglossary` to the current glossary type (normally set at the start of `\@printglossary`) and needs to define `\glscurrententrylabel` to the entry's label (normally set before `\glossentry` and `\subglossentry`). This needs some protection in case it's used in a section heading.

```

5438 \newcommand*{\glxtrglossentry}[1]{%
5439   \glxtrtitleorpdforheading
5440   {\@glxtrglossentry{#1}}%
5441   {\glsentryname{#1}}%
5442   {\glxtrheadname{#1}}%
5443 }

```

glxtrglossentry

Another test is needed in case `\@glxtrglossentry` has been written to the table of contents.

```

5444 \newrobustcmd*{\@glxtrglossentry}[1]{%
5445   \glxtrtitleorpdforheading
5446   {%
5447     \glsdodefexists{#1}%
5448     {%
5449       \begingroup
5450         \protected@edef\glscurrententrylabel{\glsetoklabel{#1}}%
5451         \protected@edef\currentglossary{\GlsXtrStandaloneGlossaryType}%
5452         \ifglshasparent{#1}%
5453         {\GlsXtrStandaloneSubEntryItem{#1}}%
5454         {\glsentryitem{#1}}%
5455         \GlsXtrStandaloneEntryName{#1}%
5456       \endgroup
5457     }%
5458   }%
5459   {\glsentryname{#1}}%
5460   {\glxtrheadname{#1}}%
5461 }

```

glxtrStandaloneEntryName

```

5462 \newcommand*{\GlsXtrStandaloneEntryName}[1]{%
5463   \glstarget{#1}{\glossentryname{#1}}%
5464 }

```

glxtrStandaloneGlossaryType

To make it easier to adjust the definition of `\currentglossary` within `\glxtrglossentry`, this expands to the default definition. (If redefined, it must fully expand to the appropriate label.)

```

5465 \newcommand{\GlsXtrStandaloneGlossaryType}{\glsentrytype{\glscurrententrylabel}}

oneSubEntryItem  Used for sub-entries in standalone format. The argument is the entry's label.
5466 \newcommand*{\GlsXtrStandaloneSubEntryItem}[1]{%
5467   \GlsXtrIfFieldEqNum{level}{#1}{1}{\glssubentryitem{#1}}{}}%
5468 }

glossentryother  As \glsxtrglossentry but uses a different field. First argument is code to use in the header.
                  The second argument is the entry's label. The third argument is the internal field label. This
                  needs to be expandable in case it occurs in a sectioning command so it can't have an optional
                  argument.
5469 \newcommand*{\glsxtrglossentryother}[3]{%
5470   \ifstrempy{#1}%
5471   {%
5472     \ifcsdef{glsxtrhead#3}%
5473     {%
5474       \glsxtrtitleorpdforheading
5475       {\@glsxtrglossentryother{#2}{#3}{#1}}%
5476       {\@gls@entry@field{#2}{#3}}%
5477       {\csuse{glsxtrhead#3}{#2}}%
5478     }%
5479     {%
5480       \glsxtrtitleorpdforheading
5481       {\@glsxtrglossentryother{#2}{#3}{#1}}%
5482       {\@gls@entry@field{#2}{#3}}%
5483       {\@gls@entry@field{\NoCaseChange{#2}}{#3}}%
5484     }%
5485   }%
5486   {%
5487     \glsxtrtitleorpdforheading
5488     {\@glsxtrglossentryother{#2}{#3}{#1}}%
5489     {\@gls@entry@field{#2}{#3}}%
5490     {#1}%
5491   }%
5492 }

glossentryother  As \@glsxtrglossentry but uses a different field.
5493 \newrobustcmd*{\@glsxtrglossentryother}[3]{%
5494   \glsxtrtitleorpdforheading
5495   {%
5496     \glsdoifexists{#1}%
5497     {%
5498       \begingroup
5499         \protected@edef\glscurrententrylabel{\glsdetoklabel{#1}}%
5500         \protected@edef\currentglossary{\GlsXtrStandaloneGlossaryType}%
5501         \ifglshasparent{#1}%
5502         {\GlsXtrStandaloneSubEntryItem{#1}}%
5503         {\glsentryitem{#1}}%

```

```

5504      \GlsXtrStandaloneEntryOther{#1}%
5505      \endgroup
5506    }%
5507  }%
5508  {\@gls@entry@field{#1}{#2}}%
5509  {#3}%
5510 }

```

`\standaloneEntryOther`

```

5511 \newcommand*{\GlsXtrStandaloneEntryOther}[2]{%
5512   \glstarget{#1}{\glossentrynameother{#1}{#2}}%
5513 }

```

`\printunsrtglossary` Similar to `\printnoidxglossary` but it displays all entries defined for the given glossary without sorting. Check for `\@printgloss@checkexists` which was introduced to glossaries v4.46.

```

5514 \ifdef\@printgloss@checkexists
5515 {
5516   \newcommand*{\printunsrtglossary}{%
5517     \let\@printgloss@checkexists\@printgloss@checkexists@allowignored
5518     \@ifstar\s@printunsrtglossary\@printunsrtglossary
5519   }
5520 }
5521 {
5522   \newcommand*{\printunsrtglossary}{%
5523     \@ifstar\s@printunsrtglossary\@printunsrtglossary
5524   }
5525 }

```

`\printunsrtglossary` Unstarred version.

```

5526 \newcommand*{\@printunsrtglossary}[1][ ]{%
5527   \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
5528 }

```

`\printunsrtglossary` Starred version.

```

5529 \newcommand*{\s@printunsrtglossary}[2][ ]{%
5530   \begingroup
5531     #2%
5532     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
5533   \endgroup
5534 }

```

`\printunsrtglossaries` Similar to `\printnoidxglossaries` but it displays all entries defined for the given glossary without sorting.

```

5535 \newcommand*{\printunsrtglossaries}{%
5536   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
5537 }

```

@unsrt@glossary

```
5538 \newcommand*{\@print@unsrt@glossary}{%
5539   \glossarysection[\glossarytoctitle]{\glossarytitle}%
5540   \glossarypreamble
5541   check for empty list
5542   {%
5543     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
5544   }%
5545   {%
5546     \key@ifundefined{glossentry}{group}%
5547     {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
5548     {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
5549     \def\@gls@currentlettergroup{}}%
```

A loop within the tabular-like styles can cause problems, so move the loop outside.

```
5550   \def\@glsxtr@doglossary{%
5551     \begin{theglossary}%
5552     \glossaryheader
5553     \glsresetentrylist
5554   }%
5555   \expandafter\@for\expandafter\@gls@currententrylabel\expandafter
5556   : \expandafter=\@csname glolist@\@glo@type\endcsname\do{%
5557     \ifdefempty{\@gls@currententrylabel}
5558     {}%
5559   }%
```

Provide a hook (for example to measure width).

```
5560     \let\@glsxtr@process\@firstofone
5561     \let\@print@unsrt@glossary@skipentry
5562     \@glsxtr@print@unsrt@glossary@skipentry
5563     \print@unsrt@glossary@entry@processhook{\@gls@currententrylabel}%
```

Don't check group for child entries.

```
5564     \glsxtr@process
5565     {%
5566       \ifglsxtr@print@gloss@groups
```

This still uses `\ifglshasparent` to determine whether or not to check for a change in the letter group. (It doesn't take the level offset into account because `bib2gls` only saves the group information for parentless entries.)

```
5567       \ifglshasparent{\@gls@currententrylabel}{}%
5568       {%
5569         \@glsxtr@checkgroup\@gls@currententrylabel
5570         \expandafter\appto\expandafter\@glsxtr@doglossary\expandafter
5571         {\@glsxtr@groupheading}%
5572       }%
5573     \fi
```

```

5574         \protected@eappto\@glxtr@doglossary{%
5575         \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
5576     }%
5577 }%
5578 }%
5579 \appto\@glxtr@doglossary{\end{theglossary}}%
5580 \printunsrtglossarypredoglossary
5581 \@glxtr@doglossary
5582 }%
5583 \glossarypostamble
5584 }

```

`\rtinnerglossary` Similar to `\printunsrtglossary` but doesn't add the section heading, preamble, postamble or start and end of theglossary. Grouping is automatically applied so it may cause a problem within tabular-like environments. The beginning and ending of theglossary should be added around this command (but ensure the style has been set first). The simplest way of doing this is to place `\printunsrtinnerglossary` inside the `\printunsrtglossarywrap` environment.

```

5585 \newcommand*{\printunsrtinnerglossary}[3][{}]{%
5586 \begingroup
5587 \def\@glxtr@printglossopts{#1}%
5588 \def\@glo@type{\gldefaulttype}%
5589 \setkeys{printgloss}[title,toctitle,style,numberedsection,sort,label]{#1}%
5590 \let\currentglossary\@glo@type
5591 #2%
5592 \@print@unsrt@innerglossary
5593 #3%
5594 \endgroup
5595 }

```

`\srtglossarywrap`

```

5596 \newenvironment{printunsrtglossarywrap}[1][{}]{%
5597 {%
5598 \def\@glxtr@printglossopts{#1}%
5599 \def\@glo@type{\gldefaulttype}%
5600 \def\glossarytitle{\csname @glotype@\@glo@type @title\endcsname}%
5601 \def\glossarytoctitle{\glossarytitle}%
5602 \let\org@glossarytitle\glossarytitle
5603 \def\@glossarystyle{%
5604 \ifx\@glossary@default@style\relax
5605 \GlossariesWarning{No default glossary style provided \MessageBreak
5606 for the glossary '@@glo@type'. \MessageBreak
5607 Using deprecated fallback. \MessageBreak
5608 To fix this set the style with \MessageBreak
5609 \string\setglossarystyle\space or use the \MessageBreak
5610 style key=value option}%
5611 \fi
5612 }%
5613 \def\gls@dotoc@title{\glssettoc@title{\@glo@type}}%
5614 \let\@org@glossaryentrynumbers\glossaryentrynumbers

```



```

5615 \@printgloss@setsort
5616 \setkeys{printgloss}{#1}%

The type key simply allows the title to be set if the title key isn't supplied.

5617 \ifglossaryexists*{\@glo@type}%
5618 {%
5619 \ifx\glossarytitle\org@glossarytitle
5620 \else
5621 \expandafter\let\csname @glo@type @title\endcsname
5622 \glossarytitle
5623 \fi
5624 \let\currentglossary\@glo@type
5625 }%
5626 {}%
5627 \let\org@glossaryentrynumbers\glossaryentrynumbers
5628 \let\glsnonextpages\@glsnonextpages
5629 \let\glsnextpages\@glsnextpages
5630 \let\nopostdesc\@nopostdesc
5631 \gls@dotoc@title
5632 \@glossarystyle
5633 \let\gls@org@glossaryentryfield\glossentry
5634 \let\gls@org@glossarysubentryfield\subglossentry

5635 \renewcommand{\glossentry}[1]{%
5636 \protected@xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
5637 \gls@org@glossaryentryfield{##1}%
5638 }%
5639 \renewcommand{\subglossentry}[2]{%
5640 \protected@xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
5641 \gls@org@glossarysubentryfield{##1}-{##2}%
5642 }%
5643 \@gls@preglossaryhook
5644 \glossarysection[\glossarytoctitle]{\glossarytitle}%
5645 \glossarypreamble
5646 \begin{theglossary}%
5647 \glossaryheader
5648 \glsresetentrylist
5649 }%
5650 {%
5651 \end{theglossary}%
5652 \glossarypostamble
5653 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
5654 \global\let\warn@noprintglossary\relax
5655 }

```

`t@innerglossary` This is much like `\@print@unsrt@innerglossary` but only contains what would normally be the content of the theglossary.

```
5656 \newcommand*{\@print@unsrt@innerglossary}{%
```

No section header or preamble.

```

5657 \glxtrifemptyglossary{\@glo@type}%
5658 {%
5659   \GlossariesExtraWarning{No entries defined in glossary '@glo@type'}%
5660 }%
5661 {%
5662   \key@ifundefined{glossentry}{group}%
5663   {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
5664   {\let\@gls@getgrouptitle\@glxtr@unsrt@getgrouptitle}%
5665   \def\@gls@currentlettergroup{}}%

```

No header or reset.

```

5666   \def\@glxtr@doglossary{%
5667   \expandafter\@for\expandafter\glscurrententrylabel\expandafter
5668     :\expandafter=\csname glolist@\@glo@type\endcsname\do{%
5669     \ifdefempty{\glscurrententrylabel}
5670     }%
5671     {%

```

Provide a hook (for example to measure width).

```

5672     \let\glxtr@process\@firstofone
5673     \let\printunsrtglossaryskipentry
5674         \@glxtr@printunsrtglossaryskipentry
5675     \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

5676     \glxtr@process
5677     {%
5678     \ifglxtr@printgloss@groups

```

This still uses \ifglshasparent to determine whether or not to check for a change in the letter group. (It doesn't take the level offset into account because bib2gls only saves the group information for parentless entries.)

```

5679         \ifglshasparent{\glscurrententrylabel}{}%
5680         {%
5681         \@glxtr@checkgroup\glscurrententrylabel
5682         \expandafter\appto\expandafter\@glxtr@doglossary\expandafter
5683         {\@glxtr@groupheading}%
5684         }%
5685     \fi

5686     \protected@eappto\@glxtr@doglossary{%
5687     \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
5688     }%
5689   }%
5690 }%
5691 \printunsrtglossarypredoglossary
5692 \@glxtr@doglossary
5693 }%

```

No postamble.

```

5694 }

```

entryprocesshook

```
5695 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{}
```

glossaryskipentry

```
5696 \newcommand*{\printunsrtglossaryskipentry}{%
5697   \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
5698 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
5699 }
```

entryprocesshook

```
5700 \newcommand*{@glxstr@printunsrtglossaryskipentry}{%
5701   \let\glxstr@process@gobble
5702 }
```

entrypredoglossary

```
5703 \newcommand*{\printunsrtglossarypredoglossary}{}%
```

glossary@handler

```
5704 \newcommand{\@printunsrt@glossary@handler}[1]{%
5705   \protected@xdef\glscurrententrylabel{#1}%
5706   \printunsrtglossaryhandler\glscurrententrylabel
5707 }
```

glossaryhandler

```
5708 \newcommand{\printunsrtglossaryhandler}[1]{%
5709   \glxstrunsrtdo{#1}%
5710 }
```

triflabelinlist

```
\glxstriflabelinlist{<label>}{<list>}{<true>}{<false>}
```

Might be useful for the handler to check if an entry label or category label is contained in a list, so provide a user-level version of `\@glx@ifinlist` which ensures the label and list are fully expanded.

```
5711 \newrobustcmd*{\glxstriflabelinlist}[4]{%
5712   \protected@edef\@glxstr@doiflabelinlist{\noexpand\@glx@ifinlist{#1}{#2}}%
5713   \@glxstr@doiflabelinlist{#3}{#4}%
5714 }
```

unsrtglossaryunit

```
5715 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
5716   \s@printunsrtglossary[type=\glstypedefaulttype,#1]{%
5717     \printunsrtglossaryunitsetup{#2}%
5718   }%
5719 }
```

ossaryunitsetup

```
5720 \newcommand*\printunsrtglossaryunitsetup}[1]{%
5721   \renewcommand{\printunsrtglossaryhandler}[1]{%
5722     \glstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
5723     {\glstrunsrtdo{##1}}}%
5724   }%
5725 }%
```

Only the target names should have the prefixes adjusted as \gls etc need the original \glolinkprefix. The \@gobble part discards \glolinkprefix.

```
5726 \ifcsundef{theH#1}%
5727 {%
5728   \renewcommand*\@glstrhypernameprefix{record.#1.\csuse{the#1}.\@gobble}%
5729 }%
5730 {%
5731   \renewcommand*\@glstrhypernameprefix{record.#1.\csuse{theH#1}.\@gobble}%
5732 }%
5733 \renewcommand*\glossarysection[2][{}]{%
5734 \appto\glossarypostamble{\glspare\medskip\glspare}%
5735 }
```

srtglossaryunit

```
5736 \newcommand{\print@noop@unsrtglossaryunit}[2][{}]{%
5737   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
5738     requires the record=only or record=alsoindex package option}{}%
5739 }
```

t@getgrouptitle

```
5740 \newrobustcmd*\@glstr@unsrt@getgrouptitle[2]{%
5741   \protected@edef\@glstr@titlelabel{\glstr@grouptitle@#1}%
5742   \@onelevel@sanitize\@glstr@titlelabel
5743   \ifcsdef{\@glstr@titlelabel}
5744     {\letcs{#2}{\@glstr@titlelabel}}%
5745     {\def#2{#1}}%
5746 }
```

\glstrunsrtdo Provide a user-level call to \@glstr@noidx@do to make it easier to define a new handler.

```
5747 \newcommand{\glstrunsrtdo}{\@glstr@noidx@do}
```

lsxtrgroupfield bib2gls provides a supplementary field labelled secondarygroup for secondary glossaries, so provide a way of switching to that field. (The group key still needs checking. There's no associated key with the internal field).

```
5748 \newcommand*\glstrgroupfield{group}
```

The tabular-like glossary styles cause quite a problem with the iterative approach. In particular for the group skip. To compensate for this, the groups are now determined while \@glstr@doglossary is being constructed rather than in the handler.

`gsxtr@checkgroup` The argument is the entry's label. (This block of code was formerly in `\@glsxtr@noidx@do`.) Now that this is no longer within a tabular environment, the global definitions aren't needed. The result is now stored in `\@glsxtr@groupheading`, which will be empty if no heading is required.

```

5749 \newcommand*{\@glsxtr@checkgroup}[1]{%
5750   \def\@glsxtr@groupheading{}%
5751   \key@ifundefined{glossentry}{group}%
5752   {%
5753     \letcs{\@gls@sort}{glo@\glsdetoklabel{#1}@sort}%
5754     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
5755   }%
5756   {%

5757     \protected@edef\@glo@thislettergrp{%
5758       \csuse{glo@\glsdetoklabel{#1}@\glsxtrgroupfield}}%
5759     }%
5760     \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
5761     {}%
5762     {%
5763       \ifdefempty{\@gls@currentlettergroup}{}%
5764       {\def\@glsxtr@groupheading{\glsgroupskip}}%
5765       \protected@eappto\@glsxtr@groupheading{%
5766         \noexpand\glsgroupheading{\expandonce\@glo@thislettergrp}%
5767       }%
5768     }%
5769     \let\@gls@currentlettergroup\@glo@thislettergrp
5770 }

```

`trLocationField` Stores the internal name of the location field.

```

5771 \newcommand*{\GlsXtrLocationField}{location}

```

`glsxtr@noidx@do` Minor modification of `\@gls@noidx@do` to check for location field if present, but also need to check for the group field.

```

5772 \newcommand{\@glsxtr@noidx@do}[1]{%
5773   \ifglsentryexists{#1}%
5774   {%
5775     \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
5776     \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@\GlsXtrLocationField}%

```

Use level number to determine whether or not this entry has a parent.

```

5777   \gls@level=\numexpr\csuse{glo@\glsdetoklabel{#1}@level}+\@glsxtr@leveloffset\relax
5778   \ifnum\gls@level>0
5779     \let\@glsxtr@ifischild\@firstoftwo
5780   \else
5781     \let\@glsxtr@ifischild\@secondoftwo
5782   \fi

```

Some glossary styles (such as `topicmcols`) save the level using `\def` so make sure `\gls@level` is expanded before being passed to `\subglossentry`.

```

5783 \@glxtr@ifischild
5784 {%
5785 \ifdefvoid{\@gls@location}%
5786 {%
5787 \ifdefvoid{\@gls@loclist}%
5788 {%
5789 \expandafter\subglossentry\expandafter{\number\gls@level}{#1}{}%
5790 }%
5791 {%
5792 \expandafter\subglossentry\expandafter{\number\gls@level}{#1}%
5793 {%
5794 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5795 }%
5796 }%
5797 }%
5798 {%
5799 \expandafter\subglossentry\expandafter
5800 {\number\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
5801 }%
5802 }%
5803 {%

5804 \ifdefvoid{\@gls@location}%
5805 {%
5806 \ifdefvoid{\@gls@loclist}
5807 {%
5808 \glossentry{#1}{}%
5809 }%
5810 {%
5811 \glossentry{#1}%
5812 {%
5813 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5814 }%
5815 }%
5816 }%
5817 {%
5818 \glossentry{#1}%
5819 {%
5820 \glossaryentrynumbers{\@gls@location}%
5821 }%
5822 }%
5823 }%
5824 }%
5825 {}%
5826 }

```

Provide a way to conveniently define commands that behaves like `\gls` with a label prefix.

It's possible that the user might want minor variations with the same prefix but different default options, so use a counter to provide unique inner commands.

\glstrnewgls

```
5827 \newcount\@glstrnewgls@inner
```

(The default options supplied in *<options>* below could possibly be used to form the inner control sequence name to help make it unique, but it might feasibly contain the value where the value might contain commands.)

r@providenewgls

```
5828 \newcommand*{\@glstr@providenewgls}{%
5829   \protected@write\@auxout{}\string\providecommand{\string\@glstr@newglslike}[2]{}%
5830   \let\@glstr@providenewgls\relax
5831 }
```

identifyglslike Identify the command given in the second argument for the benefit of **bib2gls**.

```
5832 \newcommand{\glstridentifyglslike}[2]{%
5833   \ifdefequal\@glstr@record@setting\@glstr@record@setting@off
5834   }%
5835   {%
5836     \@glstr@providenewgls
5837     \protected@write\@auxout{}\string\@glstr@newglslike{#1}\string#2}%
5838   }%
5839 }
```

\@glstrnewgls

`\glstrnewgls[<options>]{<prefix>}{<cs>}{<inner cs name>}`

```
5840 \newcommand*{\@glstrnewgls}[4]{%
5841   \ifdef{#3}%
5842   {%
5843     \PackageError{glossaries-extra}{Command \string#3\space already
5844 defined}{}%
5845   }%
5846   {%
```

Write information to the aux file for bib2gls.

```
5847   \glstridentifyglslike{#2}{#3}%
5848   \ifcsdef{@#4like@#2}%
5849   {%
5850     \advance\@glstrnewgls@inner by \@ne
5851     \def\@glstrnewgls@innercsname{@#4like\number\@glstrnewgls@inner @#2}%
5852   }%
5853   {\def\@glstrnewgls@innercsname{@#4like@#2}}%
5854   \expandafter\newrobustcmd\expandafter*\expandafter
5855     #3\expandafter{\expandafter\@glshyp@opt\csname\@glstrnewgls@innercsname\endcsname}%
5856   \ifstrempy{#1}%
5857   {%
```

```

5858 \expandafter\newcommand\expandafter*\csname\@glxstrnewgls@innercsname\endcsname[2] [] {%
5859 \new@ifnextchar[%
5860 {\csname @#4@\endcsname{##1}{#2##2}}}%
5861 {\csname @#4@\endcsname{##1}{#2##2} [] }%
5862 }%
5863 }%
5864 {%
5865 \expandafter\newcommand\expandafter*\csname\@glxstrnewgls@innercsname\endcsname[2] [] {%
5866 \new@ifnextchar[%
5867 {\csname @#4@\endcsname{#1,##1}{#2##2}}}%
5868 {\csname @#4@\endcsname{#1,##1}{#2##2} [] }%
5869 }%
5870 }%
5871 }%
5872 }

```

`\glxstrnewgls`

`\glxstrnewgls[<options>]{<prefix>}{<cs>}`

The first argument prepends to the options and the second argument is the prefix.

```

5873 \newrobustcmd*{\glxstrnewgls}[3] [] {%
5874 \@glxstrnewgls{#1}{#2}{#3}{gls}%
5875 }

```

`\glxstrnewglslike` Provide a way to conveniently define commands that behave like `\gls`, `\glspl`, `\Gls` and `\Glspl` with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```

5876 \newrobustcmd*{\glxstrnewglslike}[6] [] {%
5877 \@glxstrnewgls{#1}{#2}{#3}{gls}%
5878 \@glxstrnewgls{#1}{#2}{#4}{glspl}%
5879 \@glxstrnewgls{#1}{#2}{#5}{Gls}%
5880 \@glxstrnewgls{#1}{#2}{#6}{Glspl}%
5881 }

```

`\glxstrnewGLSlike` Provide a way to conveniently define commands that behave like `\GLS`, `\GLSpl` with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```

5882 \newrobustcmd*{\glxstrnewGLSlike}[4] [] {%
5883 \@glxstrnewgls{#1}{#2}{#3}{GLS}%
5884 \@glxstrnewgls{#1}{#2}{#4}{GLSpl}%
5885 }

```

`\glxstrnewrgls` As `\glxstrnewgls` but for `\rgls`.

```

5886 \newrobustcmd*{\glxstrnewrgls}[3] [] {%
5887 \@glxstrnewgls{#1}{#2}{#3}{rgls}%
5888 }

```


sxtrnewrglslike As \glxtrnewrglslike but for \rgls etc.

```
5889 \newrobustcmd*{\glxtrnewrglslike}[6][[]]{%
5890   \@glxtrnewrgls{#1}{#2}{#3}{rgls}%
5891   \@glxtrnewrgls{#1}{#2}{#4}{rglsp1}%
5892   \@glxtrnewrgls{#1}{#2}{#5}{rGls}%
5893   \@glxtrnewrgls{#1}{#2}{#6}{rGlspl}%
5894 }
```

sxtrnewrGLSlike As \glxtrnewGLSlike but for \rGLS etc.

```
5895 \newrobustcmd*{\glxtrnewrGLSlike}[4][[]]{%
5896   \@glxtrnewrgls{#1}{#2}{#3}{rGLS}%
5897   \@glxtrnewrgls{#1}{#2}{#4}{rGLSp1}%
5898 }
```

Provide easy access to record count fields.

totalRecordCount Access total record count. This is designed to be expandable. The argument is the label.

```
5899 \newcommand*{\GlsXtrTotalRecordCount}[1]{%
5900   \ifcsdef{glo@\glstdetoklabel{#1}@recordcount}%
5901   {\csname glo@\glstdetoklabel{#1}@recordcount\endcsname}%
5902   {0}%
5903 }
```

sXtrRecordCount Access record count for a particular counter. The first argument is the label. The second argument is the counter name.

```
5904 \newcommand*{\GlsXtrRecordCount}[2]{%
5905   \ifcsdef{glo@\glstdetoklabel{#1}@recordcount.#2}%
5906   {\csname glo@\glstdetoklabel{#1}@recordcount.#2\endcsname}%
5907   {0}%
5908 }
```

tionRecordCount Access record count for a particular counter and location. The first argument is the label. The second argument is the counter name. The third argument is the location. This command shouldn't be used if the location doesn't fully expand unless \glxtrdetoklocation can be set to something sensible.

```
5909 \newcommand*{\GlsXtrLocationRecordCount}[3]{%
5910   \ifcsdef{glo@\glstdetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}}%
5911   {\csname glo@\glstdetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}\endcsname}%
5912   {0}%
5913 }
```

trdetoklocation

```
5914 \newcommand*{\glxtrdetoklocation}[1]{#1}
```

ablerecordcount

```
5915 \newcommand*{\glxtrenablerecordcount}{%
5916   \renewcommand*{\rgls}{\rgls}%
5917   \renewcommand*{\rGls}{\rGls}%
5918 }
```

```

5918 \renewcommand*{\glspl}{\rglspl}%
5919 \renewcommand*{\Glspl}{\rGlspl}%
5920 \renewcommand*{\GLS}{\rGLS}%
5921 \renewcommand*{\GLSpl}{\rGLSpl}%
5922 }

```

ordtriggervalue The value used by the record trigger test. The argument is the entry's label.

```

5923 \newcommand*{\glsxtrrecordtriggervalue}[1]{%
5924 \GlsXtrTotalRecordCount{#1}%
5925 }

```

dCountAttribute

```

5926 \newcommand*{\GlsXtrSetRecordCountAttribute}[2]{%
5927 \@for\@glsxtr@cat:=#1\do
5928 {%
5929 \ifdefempty{\@glsxtr@cat}{}%
5930 {%
5931 \glssetcategoryattribute{\@glsxtr@cat}{recordcount}{#2}%
5932 }%
5933 }%
5934 }

```

ifrecordtrigger

$\backslash\text{glsxtrifrecordtrigger}\{\langle\text{label}\rangle\}\{\langle\text{trigger format}\rangle\}\{\langle\text{normal}\rangle\}$
--

```

5935 \newcommand*{\glsxtrifrecordtrigger}[3]{%
5936 \glsifhasattribute{#1}{recordcount}%
5937 {%
5938 \ifnum\glsxtrrecordtriggervalue{#1}>\glsgetattribute{#1}{recordcount}\relax
5939 #3%
5940 \else
5941 #2%
5942 \fi
5943 }%
5944 {#3}%
5945 }

```

strigger@record Still need a record to ensure that bib2gls selects the entry.

```

5946 \newcommand*{\@glsxtr@rglstrigger@record}[3]{%
5947 \protected@edef\glslabel{\glsdetoklabel{#2}}%
5948 \let\@gls@link@label\glslabel
5949 \def\@glsxtr@thevalue{%
5950 \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
5951 \def\@glsnumberformat{\glsstriggerrecordformat}%
5952 \protected@edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%

```

```

5953 \protected@edef\glstype{\csname glo@\glslabel @type\endcsname}%
5954 \def\@glsxtr@thevalue{%
5955 \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%

Save local setting.

5956 \@gls@save@glslocal
5957 \glsxtrinitwrgloss
5958 \glslinkpresetkeys
5959 \setkeys{glslink}{#1}%
5960 \glslinkpostsetkeys
5961 \ifdefempty{\@glsxtr@thevalue}%
5962 {%
5963 \@gls@saveentrycounter
5964 }%
5965 {%
5966 \let\theglsentrycounter\@glsxtr@thevalue
5967 \def\theHglentrycounter{\@glsxtr@theHvalue}%
5968 }%
5969 \glslinkwrcontent
5970 {%
5971 \ifglsxtrinitwrglossbefore
5972 \do@wrglossary{#2}%
5973 \fi
5974 #3%
5975 \ifglsxtrinitwrglossbefore
5976 \else
5977 \do@wrglossary{#2}%
5978 \fi
5979 }%
5980 \@gls@restore@glslocal
5981 \@gls@do@glsunset{#2}%
5982 }

```

gerrecordformat Typically won't be used as it should be recognised as a special type of ignored location by bib2gls.

```

5983 \newcommand*{\glstriggerrecordformat}[1]{

```

\rgls

```

5984 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}

```

\@rgls

```

5985 \newcommand*{\@rgls}[2][{}]{%
5986 \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}%
5987 }

```

\@rgls@

```

5988 \def\@rgls@#1#2[#3]{%
5989 \glsxtrifrecordtrigger{#2}%
5990 {%

```

```

5991 \@glxtr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
5992 }%
5993 {%
5994 \@gls@{#1}{#2}[#3]%
5995 }%
5996 }%

```

\rglspl

```

5997 \newrobustcmd*{\rglspl}{\@gls@hyp@opt\@rglspl}

```

\@rglspl

```

5998 \newcommand*{\@rglspl}[2][{}]{%
5999 \new@ifnextchar[{\@rglspl@{#1}{#2}}{\@rglspl@{#1}{#2}[]}%
6000 }

```

\@rglspl@

```

6001 \def\@rglspl@#1#2[#3]{%
6002 \glxtrifrecordtrigger{#2}%
6003 {%
6004 \@glxtr@rglstrigger@record{#1}{#2}{\rglsplformat{#2}{#3}}%
6005 }%
6006 {%
6007 \@glspl@{#1}{#2}[#3]%
6008 }%
6009 }%

```

\rGls

```

6010 \newrobustcmd*{\rGls}{\@gls@hyp@opt\@rGls}

```

\@rGls

```

6011 \newcommand*{\@rGls}[2][{}]{%
6012 \new@ifnextchar[{\@rGls@{#1}{#2}}{\@rGls@{#1}{#2}[]}%
6013 }

```

\@rGls@

```

6014 \def\@rGls@#1#2[#3]{%
6015 \glxtrifrecordtrigger{#2}%
6016 {%
6017 \@glxtr@rglstrigger@record{#1}{#2}{\rGlsformat{#2}{#3}}%
6018 }%
6019 {%
6020 \@Gls@{#1}{#2}[#3]%
6021 }%
6022 }%

```

\rGlspl

```

6023 \newrobustcmd*{\rGlspl}{\@gls@hyp@opt\@rGlspl}

```

```

\@rGlspl
6024 \newcommand*{\@rGlspl}[2] [] {%
6025   \new@ifnextchar [{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2} []}%
6026 }

\@rGlspl@
6027 \def\@rGlspl@#1#2[#3] {%
6028   \glstrifrecordtrigger{#2}%
6029   {%
6030     \@glstr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
6031   }%
6032   {%
6033     \@Glspl@{#1}{#2}[#3]%
6034   }%
6035 }%

\rGLS
6036 \newrobustcmd*{\rGLS}{\@gls@hyp@opt\rGLS}

\@rGLS
6037 \newcommand*{\@rGLS}[2] [] {%
6038   \new@ifnextchar [{\@rGLS@{#1}{#2}}{\@rGLS@{#1}{#2} []}%
6039 }

\@rGLS@
6040 \def\@rGLS@#1#2[#3] {%
6041   \glstrifrecordtrigger{#2}%
6042   {%
6043     \@glstr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%
6044   }%
6045   {%
6046     \@GLS@{#1}{#2}[#3]%
6047   }%
6048 }%

\rGLSpl
6049 \newrobustcmd*{\rGLSpl}{\@gls@hyp@opt\rGLSpl}

\@rGLSpl
6050 \newcommand*{\@rGLSpl}[2] [] {%
6051   \new@ifnextchar [{\@rGLSpl@{#1}{#2}}{\@rGLSpl@{#1}{#2} []}%
6052 }

\@rGLSpl@
6053 \def\@rGLSpl@#1#2[#3] {%
6054   \glstrifrecordtrigger{#2}%
6055   {%
6056     \@glstr@rglstrigger@record{#1}{#2}{\rGLSplformat{#2}{#3}}%

```

```

6057 }%
6058 {%
6059   \@GLSpl@{#1}{#2}[#3]%
6060 }%
6061 }%

```

`\rglsformat`

```

6062 \newcommand*{\rglsformat}[2]{%
6063   \glsifregular{#1}
6064   {\glsentryfirst{#1}}%
6065   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
6066 }

```

`\rglsplformat`

```

6067 \newcommand*{\rglsplformat}[2]{%
6068   \glsifregular{#1}
6069   {\glsentryfirstplural{#1}}%
6070   {\ifglshaslong{#1}{\glsentrylongplural{#1}}{\glsentryfirstplural{#1}}}%#2%
6071 }

```

`\rGlsformat`

```

6072 \newcommand*{\rGlsformat}[2]{%
6073   \glsifregular{#1}
6074   {\Glsentryfirst{#1}}%
6075   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
6076 }

```

`\rGlsplformat`

```

6077 \newcommand*{\rGlsplformat}[2]{%
6078   \glsifregular{#1}
6079   {\Glsentryfirstplural{#1}}%
6080   {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}}%#2%
6081 }

```

`\rGLSformat`

```

6082 \newcommand*{\rGLSformat}[2]{%
6083   \expandafter\mfirstucMakeUppercase\expandafter{\rglsformat{#1}{#2}}%
6084 }

```

`\rGLSplformat`

```

6085 \newcommand*{\rGLSplformat}[2]{%
6086   \expandafter\mfirstucMakeUppercase\expandafter{\rglsplformat{#1}{#2}}%
6087 }

```

1.4 Link Counting

This is different to the entry counting provided by the base package (which counts the number of times the first use flag is unset). Instead, this method hooks into `\@gls@link` (through

`\glxtr@inc@linkcount`) to increment an associated counter. To preserve resources, the counter is only defined if it needs to be incremented. This method is independent of the presence of hyperlinks. (The “link” part of the name refers to `\@gls@link` not `\hyperlink`.)

`\@inc@linkcount` This performs the actual incrementing and counter definition. The counter is given by `\c@glxtr@linkcount@<label>` where `<label>` is the entry’s label. Since this is performed within `\@gls@link` the label can be accessed with `\glslabel`.

```
6088 \newcommand{\@glxtr@do@inc@linkcount}{%
```

Does this entry have the linkcount attribute set?

```
6089 \glsifattribute{\glslabel}{linkcount}{true}%
```

```
6090 {%
```

Does the counter exist?

```
6091 \ifcsdef{c@glxtr@linkcount@\glslabel}{}%
```

```
6092 {%
```

Counter doesn’t exist, so define it.

```
6093 \newcounter{glxtr@linkcount@\glslabel}%
```

If linkcountmaster is set, add to counter reset.

```
6094 \glshasattribute{\glslabel}{linkcountmaster}%
```

```
6095 {%
```

Need to ensure values are fully expanded.

```
6096 \begingroup
```

```
6097 \edef\@glo@tmp{\endgroup\noexpand\@addtoreset{glxtr@linkcount@\glslabel}%
```

```
6098 {\glsggetattribute{\glslabel}{linkcountmaster}}}%
```

```
6099 \@glo@tmp
```

```
6100 }%
```

```
6101 {}%
```

```
6102 }%
```

Increment counter:

```
6103 \glxtrinlinkcounter{glxtr@linkcount@\glslabel}%
```

```
6104 }%
```

```
6105 {}%
```

```
6106 }
```

`\rlinkcounter` May be redefined to use `\refstepcounter` if required.

```
6107 \newcommand*{\glxtrinlinkcounter}[1]{\stepcounter{#1}}
```

`\linkCounterValue` Expands to the associated link counter register or 0 if not defined.

```
6108 \newcommand*{\GlsXtrLinkCounterValue}[1]{%
```

```
6109 \ifcsundef{c@glxtr@linkcount@#1}{0}{\csname c@glxtr@linkcount@#1\endcsname}%
```

```
6110 }
```

`\rTheLinkCounter` Expands to the display value of the associated link counter or 0 if not defined.

```
6111 \newcommand*{\GlsXtrTheLinkCounter}[1]{%
```

```
6112 \ifcsundef{theglsxtr@linkcount@#1}{0}%
```

```
6113 {\csname theglxtr@linkcount@#1\endcsname}%
```

```
6114 }
```

`fLinkCounterDef` Tests if the counter has been defined

```
6115 \newcommand*{\GlsXtrIfLinkCounterDef}[3]{%
6116   \ifcsundef{theglsxtr@linkcount@#1}{#3}{#2}%
6117 }
```

`LinkCounterName` Expands to the associated link counter name. (No check for existence.)

```
6118 \newcommand*{\GlsXtrLinkCounterName}[1]{\glxtr@linkcount@#1}
```

`bleLinkCounting`

`\GlsXtrEnableLinkCounting[<master counter>]{<categories>}`

Enable link counting for the given categories.

```
6119 \newcommand*{\GlsXtrEnableLinkCounting}[2][ ]{%
6120   \let\glxtr@inc@linkcount\@glxtr@do@inc@linkcount
6121   \@for\@glxtr@label:=#2\do
6122   {%
6123     \glsssetcategoryattribute{\@glxtr@label}{linkcount}{true}%
6124     \ifstrempy{#1}{}%
6125     {%
6126       \ifcsundef{c@#1}%
6127       {\@nocounterr{#1}}%
6128       {\glsssetcategoryattribute{\@glxtr@label}{linkcountmaster}{#1}}%
6129     }%
6130   }%
6131 }
6132 \@onlypreamble\GlsXtrEnableLinkCounting
```

1.5 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.

```
6133 \ifpackageloaded{glossaries-accsupp}
6134 {
```

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

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

```
6135   \newcommand*{\glsaccessname}[1]{%
6136     \glsnameaccessdisplay
6137     {%
6138       \glsentryname{#1}%
6139     }%
6140     {#1}%
6141   }
```


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

```

6142 \newcommand*{\Glsaccessname}[1]{%
6143   \glsnameaccessdisplay
6144   {%
6145     \Glsentryname{#1}%
6146   }%
6147   {#1}%
6148 }

```

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

```

6149 \newcommand*{\GLSaccessname}[1]{%
6150   \glsnameaccessdisplay
6151   {%
6152     \mfirstucMakeUppercase{\glsentryname{#1}}%
6153   }%
6154   {#1}%
6155 }

```

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

```

6156 \newcommand*{\glsaccesstext}[1]{%
6157   \glstextaccessdisplay
6158   {%
6159     \glsentrytext{#1}%
6160   }%
6161   {#1}%
6162 }

```

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

```

6163 \newcommand*{\Glsaccesstext}[1]{%
6164   \glstextaccessdisplay
6165   {%
6166     \Glsentrytext{#1}%
6167   }%
6168   {#1}%
6169 }

```

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

```

6170 \newcommand*{\GLSaccesstext}[1]{%
6171   \glstextaccessdisplay
6172   {%
6173     \mfirstucMakeUppercase{\glsentrytext{#1}}%
6174   }%
6175   {#1}%
6176 }

```

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

```

6177 \newcommand*{\glsaccessplural}[1]{%
6178   \glspluralaccessdisplay
6179   {%
6180     \glsentryplural{#1}%
6181   }%
6182   {#1}%
6183 }

```

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

```

6184 \newcommand*{\Glsaccessplural}[1]{%
6185   \glspluralaccessdisplay
6186   {%
6187     \Glsentryplural{#1}%
6188   }%
6189   {#1}%
6190 }

```

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

```

6191 \newcommand*{\GLSaccessplural}[1]{%
6192   \glspluralaccessdisplay
6193   {%
6194     \mfirstucMakeUppercase{\glsentryplural{#1}}%
6195   }%
6196   {#1}%
6197 }

```

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

```

6198 \newcommand*{\glsaccessfirst}[1]{%
6199   \glsfirstaccessdisplay
6200   {%
6201     \glsentryfirst{#1}%
6202   }%
6203   {#1}%
6204 }

```

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

```

6205 \newcommand*{\Glsaccessfirst}[1]{%
6206   \glsfirstaccessdisplay
6207   {%
6208     \Glsentryfirst{#1}%
6209   }%
6210   {#1}%
6211 }

```

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

```

6212 \newcommand*{\GLSaccessfirst}[1]{%

```

```

6213 \glsfirstaccessdisplay
6214 {%
6215 \mfirstucMakeUppercase{\glstentryfirst{#1}}%
6216 }%
6217 {#1}%
6218 }

```

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

```

6219 \newcommand*{\glsaccessfirstplural}[1]{%
6220 \glsfirstpluralaccessdisplay
6221 {%
6222 \glstentryfirstplural{#1}%
6223 }%
6224 {#1}%
6225 }

```

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

```

6226 \newcommand*{\Glsaccessfirstplural}[1]{%
6227 \glsfirstpluralaccessdisplay
6228 {%
6229 \Glstentryfirstplural{#1}%
6230 }%
6231 {#1}%
6232 }

```

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

```

6233 \newcommand*{\GLSaccessfirstplural}[1]{%
6234 \glsfirstpluralaccessdisplay
6235 {%
6236 \mfirstucMakeUppercase{\glstentryfirstplural{#1}}%
6237 }%
6238 {#1}%
6239 }

```

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

```

6240 \newcommand*{\glsaccesssymbol}[1]{%
6241 \glssymbolaccessdisplay
6242 {%
6243 \glstentrysymbol{#1}%
6244 }%
6245 {#1}%
6246 }

```

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

```

6247 \newcommand*{\Glsaccesssymbol}[1]{%
6248 \glssymbolaccessdisplay

```

```

6249   {%
6250       \Glsentrysymbol{#1}%
6251   }%
6252   {#1}%
6253 }

```

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

```

6254 \newcommand*{\GLSaccesssymbol}[1]{%
6255     \glssymbolaccessdisplay
6256     {%
6257         \mfirstucMakeUppercase{\glentrysymbol{#1}}%
6258     }%
6259     {#1}%
6260 }

```

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

```

6261 \newcommand*{\glaccesssymbolplural}[1]{%
6262     \glssymbolpluralaccessdisplay
6263     {%
6264         \glentrysymbolplural{#1}%
6265     }%
6266     {#1}%
6267 }

```

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

```

6268 \newcommand*{\Glsaccesssymbolplural}[1]{%
6269     \glssymbolpluralaccessdisplay
6270     {%
6271         \Glsentrysymbolplural{#1}%
6272     }%
6273     {#1}%
6274 }

```

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

```

6275 \newcommand*{\GLSaccesssymbolplural}[1]{%
6276     \glssymbolpluralaccessdisplay
6277     {%
6278         \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
6279     }%
6280     {#1}%
6281 }

```

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

```

6282 \newcommand*{\glsaccessdesc}[1]{%
6283     \glsdescriptionaccessdisplay
6284     {%
6285         \glentrydesc{#1}%

```

```

6286     }%
6287     {#1}%
6288 }

```

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

```

6289 \newcommand*{\Glsaccessdesc}[1]{%
6290   \glsdescriptionaccessdisplay
6291   {%
6292     \Glsentrydesc{#1}%
6293   }%
6294   {#1}%
6295 }

```

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

```

6296 \newcommand*{\GLSaccessdesc}[1]{%
6297   \glsdescriptionaccessdisplay
6298   {%
6299     \mfirstucMakeUppercase{\glsentrydesc{#1}}%
6300   }%
6301   {#1}%
6302 }

```

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

```

6303 \newcommand*{\glsaccessdescplural}[1]{%
6304   \glsdescriptionpluralaccessdisplay
6305   {%
6306     \glsentrydescplural{#1}%
6307   }%
6308   {#1}%
6309 }

```

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

```

6310 \newcommand*{\Glsaccessdescplural}[1]{%
6311   \glsdescriptionpluralaccessdisplay
6312   {%
6313     \Glsentrydescplural{#1}%
6314   }%
6315   {#1}%
6316 }

```

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

```

6317 \newcommand*{\GLSaccessdescplural}[1]{%
6318   \glsdescriptionpluralaccessdisplay
6319   {%
6320     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
6321   }%

```

```

6322     {#1}%
6323 }

```

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

```

6324 \newcommand*\glsaccessshort}[1]{%
6325   \glsshortaccessdisplay
6326   {%
6327     \glentryshort{#1}%
6328   }%
6329   {#1}%
6330 }

```

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

```

6331 \newcommand*\Glsaccessshort}[1]{%
6332   \glsshortaccessdisplay
6333   {%
6334     \Glentryshort{#1}%
6335   }%
6336   {#1}%
6337 }

```

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

```

6338 \newcommand*\GLSaccessshort}[1]{%
6339   \glsshortaccessdisplay
6340   {%
6341     \mfirstucMakeUppercase{\glentryshort{#1}}%
6342   }%
6343   {#1}%
6344 }

```

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

```

6345 \newcommand*\lsaccessshortpl}[1]{%
6346   \glsshortpluralaccessdisplay
6347   {%
6348     \glentryshortpl{#1}%
6349   }%
6350   {#1}%
6351 }

```

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

```

6352 \newcommand*\Lsaccessshortpl}[1]{%
6353   \glsshortpluralaccessdisplay
6354   {%
6355     \Glentryshortpl{#1}%
6356   }%
6357   {#1}%
6358 }

```

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

```
6359 \newcommand*{\LSaccessshortpl}[1]{%
6360   \glsshortpluralaccessdisplay
6361   {%
6362     \mfirstucMakeUppercase{\glsentryshortpl{#1}}%
6363   }%
6364   {#1}%
6365 }
```

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

```
6366 \newcommand*{\glaccesslong}[1]{%
6367   \glslongaccessdisplay{\glsentrylong{#1}}{#1}%
6368 }
```

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

```
6369
6370 \newcommand*{\Glsaccesslong}[1]{%
6371   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
6372 }
```

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

```
6373 \newcommand*{\GLSaccesslong}[1]{%
6374   \glslongaccessdisplay
6375   {%
6376     \mfirstucMakeUppercase{\glsentrylong{#1}}%
6377   }%
6378   {#1}%
6379 }
```

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

```
6380 \newcommand*{\glaccesslongpl}[1]{%
6381   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
6382 }
```

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

```
6383
6384 \newcommand*{\Glsaccesslongpl}[1]{%
6385   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
6386 }
```

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

```
6387 \newcommand*{\GLSaccesslongpl}[1]{%
6388   \glslongpluralaccessdisplay
6389   {%
6390     \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
6391   }%
6392   {#1}%
6393 }
```

Keys for accessibility support.

```
6394 \define@key{glstrabbrv}{access}{%
6395   \def\@gls@nameaccess{#1}%
6396 }
6397 \define@key{glstrabbrv}{textaccess}{%
6398   \def\@gls@textaccess{#1}%
6399 }
6400 \define@key{glstrabbrv}{pluralaccess}{%
6401   \def\@gls@pluralaccess{#1}%
6402 }
6403 \define@key{glstrabbrv}{firstaccess}{%
6404   \def\@gls@firstaccess{#1}%
6405 }
6406 \define@key{glstrabbrv}{firstpluralaccess}{%
6407   \def\@gls@firstpluralaccess{#1}%
6408 }
6409 \define@key{glstrabbrv}{shortaccess}{%
6410   \def\@gls@shortaccess{#1}%
6411 }
6412 \define@key{glstrabbrv}{shortpluralaccess}{%
6413   \def\@gls@shortaccesspl{#1}%
6414 }
6415 \define@key{glstrabbrv}{longaccess}{%
6416   \def\@gls@longaccess{#1}%
6417 }
6418 \define@key{glstrabbrv}{shortlonglaccess}{%
6419   \def\@gls@longaccesspl{#1}%
6420 }
```

@initaccesskeys

```
6421 \newcommand*{\@gls@initaccesskeys}{%
6422   \def\@gls@nameaccess{}%
6423   \def\@gls@textaccess{}%
6424   \def\@gls@pluralaccess{}%
6425   \def\@gls@firstaccess{}%
6426   \def\@gls@firstpluralaccess{}%
6427   \def\@gls@shortaccess{}%
6428   \def\@gls@shortaccesspl{}%
6429   \def\@gls@longaccess{}%
6430   \def\@gls@longaccesspl{}%
6431 }
```

ssattribute@set


```
\gls@ifaccessattribute@set{<attribute>}{<true>}{<false>}
```

```
6432 \newcommand*{\@gls@ifaccessattribute@set}[3]{%
6433   \glsifcategoryattribute{\glscategorylabel}{access#1}{true}%
6434   {#2}%
6435   {%
6436     \glsifcategoryattribute{\glscategorylabel}{access#1}{false}%
6437     {#3}%
6438     {%
6439       \glsifcategoryattribute{\glscategorylabel}{#1}{true}%
6440       {#2}%
6441       {#3}%
6442     }%
6443   }%
6444 }
```

As from glossaries v4.45, the replacement text support has been corrected so that the accessibility support for abbreviations use the “E” (expanded value) element. This should actually contain the long form since it’s supposed to explain the abbreviation. This is a bit redundant on first use for styles like long-short.

aultshortaccess

```
\glsdefaultshortaccess{<long>}{<short>}
```

This command was only introduced to glossaries-accsupp 1.42 so it may not be defined.

```
6445 \def\glsdefaultshortaccess#1#2{#1 (#2)}
```

signactualsetup

```
6446 \newcommand{\glxtrassignactualsetup}{%
6447   \let\@empty
6448   \let\emph\@firstofone
6449   \let\textbf\@firstofone
6450   \let\textmd\@firstofone
6451   \let\textit\@firstofone
6452   \let\textsl\@firstofone
6453   \let\textsc\@firstofone
6454   \let\textrm\@firstofone
6455   \let\textsf\@firstofone
6456   \let\texttt\@firstofone
6457 }
```

s@assign@actual

```
6458 \ifdef\pdfstringdef
6459 {
6460   \newcommand{\@gls@assign@actual}{%
```

```

6461 \beginngroup
6462 \glstrassignactualsetup
6463 \pdfstringdef\@gls@actualshort{\glstrorgshort}%
6464 \pdfstringdef\@gls@actuaallong{\glstrorglong}%
6465 \pdfstringdef\@gls@actualshortpl{\@gls@shortpl}%
6466 \pdfstringdef\@gls@actuaallongpl{\@gls@longpl}%
6467 \protected@edef\@gls@tmp{\endgroup
6468 \def\noexpand\@gls@actualshort{\expandonce\@gls@actualshort}%
6469 \def\noexpand\@gls@actuaallong{\expandonce\@gls@actuaallong}%
6470 \def\noexpand\@gls@actualshortpl{\expandonce\@gls@actualshortpl}%
6471 \def\noexpand\@gls@actuaallongpl{\expandonce\@gls@actuaallongpl}%
6472 }%
6473 \@gls@tmp
6474 }
6475 }
6476 {
6477 \newcommand{\@gls@assign@actual}{%
6478 \beginngroup
6479 \glstrassignactualsetup
6480 \protected@edef\@gls@tmp{\endgroup
6481 \def\noexpand\@gls@actualshort{\glstrorgshort}%
6482 \def\noexpand\@gls@actuaallong{\glstrorglong}%
6483 \def\noexpand\@gls@actualshortpl{\@gls@shortpl}%
6484 \def\noexpand\@gls@actuaallongpl{\@gls@longpl}%
6485 }%
6486 \@gls@tmp
6487 }
6488 }

```

`\lt@short@access` Renamed `\@gls@setup@default@access` and removed argument since it can be obtained from `\glstrorgshort`.

`@default@access` Assign the default value of the shortaccess key. The argument is the short value passed to `\newabbreviation`. The shortaccess value should explain the abbreviation.

```

6489 \newcommand{\@gls@setup@default@access}{%
6490 \@gls@assign@actual
6491 \ifdefempty\@gls@shortaccess
6492 {%

```

Check if the `accessinsertdots` attribute has been set but only if `shortaccess` hasn't been set.

```

6493 \@gls@ifaccessattribute@set{insertdots}%
6494 {%
6495 \expandafter\@glstr@insertdots\expandafter\@gls@actualshort\expandafter
6496 {\@gls@actualshort}%
6497 }%
6498 {}%
6499 \ifdefempty\@gls@longaccess
6500 {%
6501 \protected@edef\@gls@shortaccess{\glsdefaultshortaccess

```

```

6502         {\expandonce\@gls@actuallong}{\expandonce\@gls@actualshort}}}%
6503     }%
6504     {%
6505         \protected@edef\@gls@shortaccess{\glsdefaultshortaccess
6506             {\expandonce\@gls@longaccess}{\expandonce\@gls@actualshort}}}%
6507     }%
6508     \eappto\ExtraCustomAbbreviationFields{shortaccess={\@gls@shortaccess},}%

```

If shortaccessplural hasn't been set, assign plural form.

```

6509     \ifdefempty\@gls@shortaccesspl
6510     {%
6511         \@gls@ifaccessattribute@set{aposplural}%
6512         {%
6513             \expandafter\def\expandafter\@gls@shortaccesspl\expandafter{%
6514                 \@gls@actualshort'\glsxtrabbrvpluralsuffix}%
6515             }%
6516             {%
6517                 \@gls@ifaccessattribute@set{noshortplural}%
6518                 {%
6519                     \let\@gls@shortaccesspl\@gls@shortaccess
6520                     }%
6521                     {%
6522                         \let\@gls@shortaccesspl\@gls@actualshortpl
6523                         }%
6524                     }%
6525                 \ifdefempty\@gls@longaccesspl
6526                 {%
6527                     \protected@edef\@gls@shortaccesspl{\glsdefaultshortaccess
6528                         {\expandonce\@gls@actuallongpl}{\expandonce\@gls@actualshortpl}}}%
6529                 }%
6530                 {%
6531                     \protected@edef\@gls@shortaccesspl{\glsdefaultshortaccess
6532                         {\expandonce\@gls@longaccesspl}{\expandonce\@gls@actualshort}}}%
6533                 }%
6534                 \eappto\ExtraCustomAbbreviationFields{shortpluralaccess={\@gls@shortaccesspl},}%
6535                 }%
6536                 {}%
6537             }%
6538             {%
6539                 \ifdefempty\@gls@shortaccesspl
6540                 {\let\@gls@shortaccesspl\@gls@shortaccess}%
6541                 {}%
6542             }%

```

If access key hasn't been set, check if the nameshortaccess attribute has been set.

```

6543     \ifdefempty\@gls@nameaccess
6544     {%
6545         \glsifcategoryattribute{\glscategorylabel}{nameshortaccess}{true}%
6546         {%

```

```

6547     \eappto\ExtraCustomAbbreviationFields{access={\@gls@shortaccess},}%
6548     }%
6549     {}%
6550     }%
6551     {}%

```

If textaccess key hasn't been set, check if the textshortaccess attribute has been set.

```

6552     \ifdefempty\@gls@textaccess
6553     {%
6554         \glsifcategoryattribute{\glscategorylabel}{textshortaccess}{true}%
6555         {%
6556             \eappto\ExtraCustomAbbreviationFields{textaccess={\@gls@shortaccess},}%
6557             }%
6558             {}%
6559         }%
6560     }%
6561     \ifdefempty\@gls@pluralaccess
6562     {%
6563         \glsifcategoryattribute{\glscategorylabel}{textshortaccess}{true}%
6564         {%
6565             \eappto\ExtraCustomAbbreviationFields{%
6566                 pluralaccess={\@gls@shortaccesspl},%
6567             }%
6568             }%
6569             {}%
6570         }%
6571     }%

```

If firstaccess key hasn't been set, check if the firstshortaccess attribute has been set.

```

6572     \ifdefempty\@gls@firstaccess
6573     {%
6574         \glsifcategoryattribute{\glscategorylabel}{firstshortaccess}{true}%
6575         {%
6576             \eappto\ExtraCustomAbbreviationFields{firstaccess={\@gls@shortaccess},}%
6577             }%
6578             {}%
6579         }%
6580     }%
6581     \ifdefempty\@gls@firstpluralaccess
6582     {%
6583         \glsifcategoryattribute{\glscategorylabel}{firstshortaccess}{true}%
6584         {%
6585             \eappto\ExtraCustomAbbreviationFields{%
6586                 firstpluralaccess={\@gls@shortaccesspl},%
6587             }%
6588             }%
6589             {}%
6590         }%
6591     }%
6592 }

```

Provide hooks for `\setabbreviationstyle` that automatically set the attributes appropriate for the style. If the name is just the short form and the description contains the long form, then it may not be necessary to set `nameshortaccess` but it would depend on the glossary style.

Need to provide `\glxtr<category><field>accsupp` if not already defined.

`provideaccsuppcmd`

```
6593 \newcommand*{\glxtrprovideaccsuppcmd}[2]{%
6594   \ifcsundef{glxtr#1#2accsupp}%
6595   {\csdef{glxtr#1#2accsupp}{\glsshortaccsupp}}%
6596   }%
6597 }
```

`rSetNoLongAttrs` For styles where the name, first and text are just the abbreviation.

```
6598 \newcommand*{\glxtrAccSuppAbbrSetNoLongAttrs}[1]{%
6599   \glssetcategoryattribute{#1}{nameshortaccess}{true}%
6600   \glssetcategoryattribute{#1}{firstshortaccess}{true}%
6601   \glssetcategoryattribute{#1}{textshortaccess}{true}%
6602   \glxtrprovideaccsuppcmd{#1}{name}%
6603   \glxtrprovideaccsuppcmd{#1}{first}%
6604   \glxtrprovideaccsuppcmd{#1}{firstpl}%
6605   \glxtrprovideaccsuppcmd{#1}{text}%
6606   \glxtrprovideaccsuppcmd{#1}{plural}%
6607 }
```

`tFirstLongAttrs` For styles where the name and text are just the abbreviation. The first form may just be long or may be short and long.

```
6608 \newcommand*{\glxtrAccSuppAbbrSetFirstLongAttrs}[1]{%
6609   \glssetcategoryattribute{#1}{nameshortaccess}{true}%
6610   \glssetcategoryattribute{#1}{textshortaccess}{true}%
6611   \glxtrprovideaccsuppcmd{#1}{name}%
6612   \glxtrprovideaccsuppcmd{#1}{text}%
6613   \glxtrprovideaccsuppcmd{#1}{plural}%
6614 }
```

`tTextShortAttrs` For styles where only the text is just the abbreviation. The name and first form may just be long or may be short and long. The name may also be short but followed by the long form in the description.

```
6615 \newcommand*{\glxtrAccSuppAbbrSetTextShortAttrs}[1]{%
6616   \glssetcategoryattribute{#1}{textshortaccess}{true}%
6617   \glxtrprovideaccsuppcmd{#1}{text}%
6618   \glxtrprovideaccsuppcmd{#1}{plural}%
6619 }
```

`tNameShortAttrs` For styles where only the name is just the abbreviation. The first and subsequent form may just be long or may be short and long.

```
6620 \newcommand*{\glxtrAccSuppAbbrSetNameShortAttrs}[1]{%
6621   \glssetcategoryattribute{#1}{nameshortaccess}{true}%
6622   \glxtrprovideaccsuppcmd{#1}{name}%
6623 }
```

etNameLongAttrs For styles where the first and text are just the abbreviation. The name may just be long or may be short and long or the name may be short.

```

6624 \newcommand*{\glxtrAccSuppAbbrSetNameLongAttrs}[1]{%
6625 \glsetcategoryattribute{#1}{firstshortaccess}{true}%
6626 \glsetcategoryattribute{#1}{textshortaccess}{true}%
6627 \glxtrprovideaccsuppcmd{#1}{first}%
6628 \glxtrprovideaccsuppcmd{#1}{firstpl}%
6629 \glxtrprovideaccsuppcmd{#1}{text}%
6630 \glxtrprovideaccsuppcmd{#1}{plural}%
6631 }

```

End of if accsupp part

```

6632 }
6633 {

```

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

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

```

6634 \newcommand*{\glsaccessname}[1]{\glentryname{#1}}

```

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

```

6635 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

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

```

6636 \newcommand*{\GLSaccessname}[1]{%
6637 \protect\mfirstucMakeUppercase{\glentryname{#1}}}

```

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

```

6638 \newcommand*{\glsaccesstext}[1]{\glentrytext{#1}}

```

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

```

6639 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}

```

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

```

6640 \newcommand*{\GLSaccesstext}[1]{%
6641 \protect\mfirstucMakeUppercase{\glentrytext{#1}}}

```

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

```

6642 \newcommand*{\glsaccessplural}[1]{\glentryplural{#1}}

```

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

```

6643 \newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}

```

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

```

6644 \newcommand*{\GLSaccessplural}[1]{%
6645 \protect\mfirstucMakeUppercase{\glentryplural{#1}}}

```

`\glsaccessfirst` Display the first value (no link and no check for existence).
6646 `\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.
6647 `\newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}`

`\GLSaccessfirst` Display the first value (no link and no check for existence). converted to upper case.
6648 `\newcommand*{\GLSaccessfirst}[1]{%`
6649 `\protect\mfirstucMakeUppercase{\glsentryfirst{#1}}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence).
6650 `\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.
6651 `\newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.
6652 `\newcommand*{\GLSaccessfirstplural}[1]{%`
6653 `\protect\mfirstucMakeUppercase{\glsentryfirstplural{#1}}}`

`glsaccesssymbol` Display the symbol value (no link and no check for existence).
6654 `\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.
6655 `\newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}`

`GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.
6656 `\newcommand*{\GLSaccesssymbol}[1]{%`
6657 `\protect\mfirstucMakeUppercase{\glsentrysymbol{#1}}}`

`esssymbolplural` Display the symbolplural value (no link and no check for existence).
6658 `\newcommand*{\glsaccesssymbolplural}[1]{\glsentrysymbolplural{#1}}`

`esssymbolplural` Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.
6659 `\newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}`

`esssymbolplural` Display the symbolplural value (no link and no check for existence). converted to upper case.
6660 `\newcommand*{\GLSaccesssymbolplural}[1]{%`
6661 `\protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}`

`\glsaccessdesc` Display the desc value (no link and no check for existence).
6662 `\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.
6663 `\newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence). converted to upper case.
6664 `\newcommand*{\GLSaccessdesc}[1]{%`
6665 `\protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}`

`ccessdescplural` Display the descplural value (no link and no check for existence).
6666 `\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.
6667 `\newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}`

`ccessdescplural` Display the descplural value (no link and no check for existence). converted to upper case.
6668 `\newcommand*{\GLSaccessdescplural}[1]{%`
6669 `\protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}`

`\glsaccessshort` Display the short form (no link and no check for existence).
6670 `\newcommand*{\glsaccessshort}[1]{\glsentryshort{#1}}`

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).
6671 `\newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}`

`\GLSaccessshort` Display the short value (no link and no check for existence). converted to upper case.
6672 `\newcommand*{\GLSaccessshort}[1]{%`
6673 `\protect\mfirstucMakeUppercase{\glsentryshort{#1}}}`

`lsaccessshortpl` Display the short plural form (no link and no check for existence).
6674 `\newcommand*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}`

`lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).
6675 `\newcommand*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}`

`LSaccessshortpl` Display the shortplural value (no link and no check for existence). converted to upper case.
6676 `\newcommand*{\GLSaccessshortpl}[1]{%`
6677 `\protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}`

`\glsaccesslong` Display the long form (no link and no check for existence).
6678 `\newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}`

`\Glsaccesslong` Display the long form (no link and no check for existence).
6679 `\newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}`

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

```

6680 \newcommand*{\GLSaccesslong}[1]{%
6681 \protect\mfirstucMakeUppercase{\glstentrylong{#1}}}
```

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

```

6682 \newcommand*{\glsaccesslongpl}[1]{\glstentrylongpl{#1}}
```

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

```

6683 \newcommand*{\Glsaccesslongpl}[1]{\Glstentrylongpl{#1}}
```

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

```

6684 \newcommand*{\GLSaccesslongpl}[1]{%
6685 \protect\mfirstucMakeUppercase{\glstentrylongpl{#1}}}
```

`@initaccesskeys` This does nothing if there's no accessibility support.

```

6686 \newcommand*{\@gls@initaccesskeys}{}
```

`@default@access` This does nothing if there's no accessibility support.

```

6687 \newcommand*{\@gls@setup@default@access}{}
```

`rSetNoLongAttrs` This does nothing if there's no accessibility support.

```

6688 \newcommand*{\glstxtrAccSuppAbbrSetNoLongAttrs}[1]{}
```

`tFirstLongAttrs` This does nothing if there's no accessibility support.

```

6689 \newcommand*{\glstxtrAccSuppAbbrSetFirstLongAttrs}[1]{}
```

`tTextShortAttrs` This does nothing if there's no accessibility support.

```

6690 \newcommand*{\glstxtrAccSuppAbbrSetTextShortAttrs}[1]{}
```

`tNameShortAttrs` This does nothing if there's no accessibility support.

```

6691 \newcommand*{\glstxtrAccSuppAbbrSetNameShortAttrs}[1]{}
```

`etNameLongAttrs` This does nothing if there's no accessibility support.

```

6692 \newcommand*{\glstxtrAccSuppAbbrSetNameLongAttrs}[1]{}
```

End of else part

```

6693 }
```

1.6 Categories

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

```

6694 \glsaddstoragekey{category}{general}{\glscategory}
```

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

```

6695 \newcommand*{\glsifcategory}[4]{%
6696 \ifglstfield{#1}{category}{#2}{#3}{#4}%
6697 }
```

Categories can have attributes.

categoryattribute

```
\glsssetcategoryattribute{<category>}{<attribute-label>}{<value>}
```

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

```
6698 \newcommand*{\glsssetcategoryattribute}[3]{%
6699   \csdef{@glssxtr@categoryattr@#1@#2}{#3}%
6700 }
```

categoriesattribute

```
\glsssetcategoriesattribute{<category list>}{<attribute-label>}{<value>}
```

Similar to above, but globally apply to each category in the list.

```
6701 \newcommand*{\glsssetcategoriesattribute}[3]{%
6702   \@for{@glss@thiscatlabel:=#1\do{%
6703     \csgdef{@glssxtr@categoryattr@@@glss@thiscatlabel @#2}{#3}%
6704   }}%
6705 }
```

categoriesattributes

```
\glsssetcategoriesattributes{<category list>}{<attribute-label list>}{<value>}
```

Similar to above, but apply to each category and attribute in the list.

```
6706 \newcommand*{\glsssetcategoriesattributes}[3]{%
```

Group to avoid problems with nested \@for.

```
6707 {%
6708   \@for{@glss@thisattrlabel:=#2\do{%
6709     \glsssetcategoriesattribute{#1}{\@glss@thisattrlabel}{#3}%
6710   }}%
6711 }%
6712 }
```

categoryattribute

```
\glssgetcategoryattribute{<category>}{<attribute-label>}
```

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

```

6713 \newcommand*{\glsgetcategoryattribute}[2]{%
6714   \csuse{@glsxtr@categoryattr@#1@#2}%
6715 }

```

categoryattribute

```
\glsunsetcategoryattribute{<category>}{<attribute-label>}
```

Unsets the given attribute for the given category.

```

6716 \newcommand*{\glsunsetcategoryattribute}[2]{%
6717   \csundef{@glsxtr@categoryattr@#1@#2}%
6718 }

```

categoryattribute

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

Tests if the category has the given attribute set.

```

6719 \newcommand*{\glshascategoryattribute}[4]{%
6720   \ifcsvoid{@glsxtr@categoryattr@#1@#2}{#4}{#3}%
6721 }

```

glssetattribute

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

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

```

6722 \newcommand*{\glssetattribute}[3]{%
6723   \glssetcategoryattribute{\glscategory{#1}}{#2}{#3}%
6724 }

```

glsgetattribute

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

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

```

6725 \newcommand*{\glsgetattribute}[2]{%
6726   \glsgetcategoryattribute{\glscategory{#1}}{#2}%
6727 }

```

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.

```
6728 \newcommand*{\glshasattribute}[4]{%
6729   \ifglentryexists{#1}%
6730   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
6731   {#4}}%
6732 }
```

categoryattribute

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

True if category has the attribute with the given value.

```
6733 \newcommand{\glcifcategoryattribute}[5]{%
6734   \ifcsundef{@glxtr@categoryattr@@#1@#2}%
6735   {#5}%
6736   {\ifcsstring{@glxtr@categoryattr@@#1@#2}{#3}{#4}{#5}}%
6737 }
```

\glcifattribute

```
\glcifattribute{<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.

```
6738 \newcommand{\glcifattribute}[5]{%
6739   \ifglentryexists{#1}%
6740   {\glcifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
6741   {#5}%
6742 }
```

Set attributes for the default general category:

```
6743 \glissetcategoryattribute{general}{regular}{true}
```

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

```
6744 \glissetcategoryattribute{acronym}{regular}{true}
```

regularcategory Convenient shortcut to add the regular attribute.

```
6745 \newcommand*{\glsetregularcategory}[1]{%
6746   \glissetcategoryattribute{#1}{regular}{true}%
6747 }
```

regularcategory

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

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

```
6748 \newcommand{\glsifregularcategory}[3]{%  
6749   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%  
6750 }
```

regularcategory

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

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

```
6751 \newcommand{\glsifnotregularcategory}[3]{%  
6752   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%  
6753 }
```

\glsifregular

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

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

```
6754 \newcommand{\glsifregular}[3]{%  
6755   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%  
6756 }
```

glsifnotregular

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

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

```
6757 \newcommand{\glsifnotregular}[3]{%  
6758   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%  
6759 }
```

foreachincategory

```
\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.

```

6760 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
6761   \forallglossaries[#1]{#3}%
6762   {%
6763     \forlentries[#3]{#4}%
6764     {%
6765       \glsifcategory{#4}{#2}{#5}{}%
6766     }%
6767   }%
6768 }

```

chwithattribute

```

\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.

```

6769 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
6770   \forallglossaries[#1]{#4}%
6771   {%
6772     \forlentries[#4]{#5}%
6773     {%
6774       \glsifattribute{#5}{#2}{#3}{#6}{}%
6775     }%
6776   }%
6777 }

```

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

```

6778 \ifdef\newterm
6779 {%

```

`\newterm`

```

6780   \renewcommand*{\newterm}[2][ ]{%
6781     \newglossaryentry{#2}%
6782     {type={index},category=index,name={#2},%
6783      description={\glstrpostdescription\nopostdesc},#1}%
6784   }

```

Indexed terms are regular by default.

```

6785   \glsssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```
6786 \newcommand*{\glxtrpostdescindex}{}  
6787 }  
6788 {}
```

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.

```
6789 \ifdef\printsymbols  
6790 {%
```

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

```
6791 \newcommand*{\glxtrnewsymbol}[3] [] {%  
6792 \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%  
6793 }
```

Symbols are regular by default.

```
6794 \glsssetcategoryattribute{symbol}{regular}{true}
```

rpostdescsymbol

```
6795 \newcommand*{\glxtrpostdescsymbol}{}  
6796 }  
6797 {}
```

Similar for the numbers option.

```
6798 \ifdef\printnumbers  
6799 {%
```

glxtrnewnumber

```
6800 \ifdef\printnumbers  
6801 \newcommand*{\glxtrnewnumber}[3] [] {%  
6802 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%  
6803 }
```

Numbers are regular by default.

```
6804 \glsssetcategoryattribute{number}{regular}{true}
```

rpostdescnumber

```
6805 \newcommand*{\glxtrpostdescnumber}{}  
6806 }  
6807 {}
```

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

```
6808 \newcommand*{\glstrsetcategory}[2]{%
6809   \@for\@glstr@label:=#1\do
6810   {%
6811     \glsfieldxdef{\@glstr@label}{category}{#2}%
6812   }%
6813 }
```

`\categoryforall` Set the category for all entries in the listed glossaries. The first argument is the list of glossary labels and the second argument is the category label.

```
6814 \newcommand*{\glstrsetcategoryforall}[2]{%
6815   \forallglossaries[#1]{\@glstr@type}{%
6816     \forallsentries[\@glstr@type]{\@glstr@label}%
6817     {%
6818       \glsfieldxdef{\@glstr@label}{category}{#2}%
6819     }%
6820   }%
6821 }
```

`\glstrfieldtitlecase`

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

Apply title casing to the contents of the given field.

```
6822 \newcommand*{\glstrfieldtitlecase}[2]{%
6823   \expandafter\glstrfieldtitlecasecs\expandafter
6824   {\cename glo@\glsdetoklabel{#1}@#2\endcename}%
6825 }
```

`\fieldtitlecasecs` The command used by `\glstrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`. Check for `\glscapitalisewords`, which was added to glossaries v4.48.

```
6826 \ifdef\glscapitalisewords
6827 {
6828   \newcommand*{\glstrfieldtitlecasecs}[1]{%
6829     \expandafter\glscapitalisewords\expandafter{#1}}
6830 }
6831 {
6832   \newcommand*{\glstrfieldtitlecasecs}[1]{\xcapitalisewords{#1}}
6833 }
```

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.


```

6834 \@ifpackageloaded{glossaries-accsupp}
6835 {
6836   \renewcommand*{\glossentrydesc}[1]{%
6837     \glsdoifexistsorwarn{#1}%
6838     {%
6839       \glssetabbrvfmt{\glscategory{#1}}%

```

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

```

6840       \glshasattribute{#1}{glossdescfont}%
6841       {%
6842         \protected@edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
6843         \ifcsdef{\@glxtr@attrval}%
6844         {%
6845           \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
6846          }%
6847         {%
6848           \GlossariesExtraWarning{Unknown control sequence name
6849             '\@glxtr@attrval' supplied in glossdescfont attribute
6850             for entry '#1'. Ignoring}%
6851           \let\@glxtr@glossdescfont\@firstofone
6852          }%
6853        }%
6854         {\let\@glxtr@glossdescfont\@firstofone}%
6855         \glsifattribute{#1}{glossdesc}{firstuc}%
6856         {%
6857           \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
6858          }%
6859         {%
6860           \glsifattribute{#1}{glossdesc}{title}%
6861           {%
6862             \@glxtr@do@titlecaps@warn
6863             \glsdescriptionaccessdisplay
6864             {%
6865               \@glxtr@glossdescfont{\@glxtr@fieldtitlecase{#1}{desc}}%
6866              }%
6867             {#1}%
6868            }%
6869           {%
6870             \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
6871            }%
6872          }%
6873        }%
6874       }
6875 }
6876 {
6877   \renewcommand*{\glossentrydesc}[1]{%
6878     \glsdoifexistsorwarn{#1}%
6879     {%
6880       \glssetabbrvfmt{\glscategory{#1}}%
6881       \glshasattribute{#1}{glossdescfont}%

```

```

6882      {%
6883          \protected@edef\@glxtr@attrval{\glsetattribute{#1}{glossdescfont}}%
6884          \ifcsdef{\@glxtr@attrval}%
6885          {%
6886              \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
6887          }%
6888          {%
6889              \GlossariesExtraWarning{Unknown control sequence name
6890              ‘\@glxtr@attrval’ supplied in glossdescfont attribute
6891              for entry ‘#1’. Ignoring}%
6892              \let\@glxtr@glossdescfont\@firstofone
6893          }%
6894      }%
6895      {\let\@glxtr@glossdescfont\@firstofone}%
6896      \glsifattribute{#1}{glossdesc}{firstuc}%
6897      {%
6898          \@glxtr@glossdescfont{\Glsentrydesc{#1}}%
6899      }%
6900      {%
6901          \glsifattribute{#1}{glossdesc}{title}%
6902          {%
6903              \@glxtr@do@titlecaps@warn
6904              \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
6905          }%
6906          {%
6907              \@glxtr@glossdescfont{\glentrydesc{#1}}%
6908          }%
6909      }%
6910  }%
6911  }
6912 }

```

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

```

6913 \ifpackageloaded{glossaries-accsupp}
6914 {
6915     \renewcommand*{\glossentryname}[1]{%
6916         \glsdoifexistsorwarn{#1}%
6917         {%
6918             \glsetabbrvfmt{\glscategory{#1}}%

```

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

```

6919         \glshasattribute{#1}{glossnamefont}%
6920         {%
6921             \protected@edef\@glxtr@attrval{\glsetattribute{#1}{glossnamefont}}%
6922             \ifcsdef{\@glxtr@attrval}%
6923             {%
6924                 \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
6925             }%

```

```

6926      {%
6927      \GlossariesExtraWarning{Unknown control sequence name
6928      '\@glsxtr@attrval' supplied in glossnamefont attribute
6929      for entry '#1'. Reverting to default \string\glsnamefont}%
6930      \let\@glsxtr@glossnamefont\glsnamefont
6931      }%
6932    }%
6933    {\let\@glsxtr@glossnamefont\glsnamefont}%
6934    \glsifattribute{#1}{glossname}{firstuc}%
6935    {%
6936      \glsnameaccessdisplay
6937      {%
6938        \@glsxtr@glossnamefont{\Glsentryname{#1}}%
6939      }%
6940      {#1}%
6941    }%
6942    {%
6943      \glsifattribute{#1}{glossname}{title}%
6944      {%
6945        \@glsxtr@do@titlecaps@warn
6946        \glsnameaccessdisplay
6947        {%
6948          \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
6949        }%
6950        {#1}%
6951      }%
6952      {%
6953        \glsifattribute{#1}{glossname}{uc}%
6954        {%
6955          \glsnameaccessdisplay
6956          {%

```

Hide the label from the upper-casing command.

```

6957      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
6958      \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
6959    }%
6960    {#1}%
6961  }%
6962  {%
6963    \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
6964    \glsnameaccessdisplay
6965    {%
6966      \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
6967    }%
6968    {#1}%
6969  }%
6970  }%
6971  }%

```

Do post-name hook:

```

6972     \glxtrpostnamehook{#1}%
6973 }%
6974 }
6975 }
6976 {
6977   \renewcommand*{\glossentryname}[1]{%
6978     \@glsdoifexistsorwarn{#1}%
6979     {%
6980       \glssetabbrvfmt{\glscategory{#1}}%
6981       \glsattribute{#1}{glossnamefont}%
6982       {%
6983         \protected@edef\@glxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
6984         \ifcsdef{\@glxtr@attrval}%
6985         {%
6986           \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
6987         }%
6988         {%
6989           \GlossariesExtraWarning{Unknown control sequence name
6990             '\@glxtr@attrval' supplied in glossnamefont attribute
6991             for entry '#1'. Reverting to default \string\glsnamefont}%
6992           \let\@glxtr@glossnamefont\glsnamefont
6993         }%
6994       }%
6995       {\let\@glxtr@glossnamefont\glsnamefont}%
6996       \glsifattribute{#1}{glossname}{firstuc}%
6997       {%
6998         \@glxtr@glossnamefont{\Glsentryname{#1}}%
6999       }%
7000     }%
7001     \glsifattribute{#1}{glossname}{title}%
7002     {%
7003       \@glxtr@do@titlecaps@warn
7004       \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
7005     }%
7006     {%
7007       \glsifattribute{#1}{glossname}{uc}%
7008     }%

```

Hide the label from the upper-casing command.

```

7009     \letcs{\glo@name}{glo\glsdetoklabel{#1}@name}%
7010     \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
7011   }%
7012   {%

```

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.

```

7013     \letcs{\glo@name}{glo\glsdetoklabel{#1}@name}%
7014     \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
7015   }%

```

```

7016         }%
7017     }%
    Do post-name hook.
7018     \glstrpostnamehook{#1}%
7019 }%
7020 }
7021 }

```

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

```

7022 \ifpackageloaded{glossaries-accsupp}
7023 {
7024   \renewcommand*{\Glossentryname}[1]{%
7025     \@glsdoifexistsorwarn{#1}%
7026     {%
7027       \glsetabbrvfmt{\glscategory{#1}}%
    As from version 1.04, allow the glossnamefont attribute to determine the font applied.
7028       \glshasattribute{#1}{glossnamefont}%
7029       {%
7030         \protected@edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
7031         \ifcsdef{\@glstr@attrval}%
7032         {%
7033           \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
7034         }%
7035         {%
7036           \GlossariesExtraWarning{Unknown control sequence name
7037             '\@glstr@attrval' supplied in glossnamefont attribute
7038             for entry '#1'. Reverting to default \string\glnamefont}%
7039           \let\@glstr@glossnamefont\glnamefont
7040         }%
7041       }%
7042       {\let\@glstr@glossnamefont\glnamefont}%
7043       \glnameaccessdisplay
7044       {%
7045         \@glstr@glossnamefont{\Glsentryname{#1}}%
7046       }%
7047       {#1}%

```

Do post-name hook:

```

7048     \glstrpostnamehook{#1}%
7049 }%
7050 }
7051 }
7052 {
7053   \renewcommand*{\Glossentryname}[1]{%
7054     \@glsdoifexistsorwarn{#1}%
7055     {%
7056       \glsetabbrvfmt{\glscategory{#1}}%
7057       \glshasattribute{#1}{glossnamefont}%

```

```

7058      {%
7059          \protected@edef\@glstr@attrval{\glstrgetattribute{#1}{glossnamefont}}%
7060          \ifcsdef{\@glstr@attrval}%
7061          {%
7062              \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
7063          }%
7064          {%
7065              \GlossariesExtraWarning{Unknown control sequence name
7066              ‘\@glstr@attrval’ supplied in glossnamefont attribute
7067              for entry ‘#1’. Reverting to default \string\glstrnamefont}%
7068              \let\@glstr@glossnamefont\glstrnamefont
7069          }%
7070      }%
7071      {\let\@glstr@glossnamefont\glstrnamefont}%
7072      \@glstr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

7073      \glstrpostnamehook{#1}%
7074      }%
7075  }
7076 }

```

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

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

```

7077 \newcommand*{\glstrpostnamehook}[1]{%
7078   \let\@glstrnumberformat\@glstr@defaultnumberformat
7079   \glstrdoautoindexname{#1}{indexname}%

```

Allow additional code regardless of category:

```

7080   \glsextrapostnamehook{#1}%

```

Allow categories to hook in here.

```

7081   \csuse{glstrpostname\glscategory{#1}}%
7082 }

```

`\glstrpostnamehook`

```

7083 \newcommand*{\glsextrapostnamehook}[1]{}%

```

`\glstrdefpostname` Provide a convenient command for defining the post-name hook for the given category.

```

7084 \newcommand*{\glstrdefpostname}[2]{%
7085   \csdef{glstrpostname#1}{#2}%
7086 }

```

`\glstraccessdisplay`

```

7087 \ifpackageloaded{glossaries-accsupp}
7088 {

```

```

7089 \newcommand*{\glxtr@setaccessdisplay}[1]{%
7090   \ifcsdef{gls#1accessdisplay}%
7091   {\letcs\@glxtr@accessdisplay{gls#1accessdisplay}}%
7092   {%

```

This is essentially the reverse of `\@gls@fetchfield`, since the field supplied to `\glossentryname` has to be the internal label, but the `\gls{field}` accessdisplay commands use the key name.

```

7093   \protected@edef\@gls@thisval{#1}%
7094   \@for\@gls@map:=\@gls@keymap\do{%
7095     \protected@edef\@this@key{\expandafter\@secondoftwo\@gls@map}%
7096     \ifdefequal{\@this@key}{\@gls@thisval}%
7097     {%
7098       \protected@edef\@gls@thisval{\expandafter\@firstoftwo\@gls@map}%
7099       \@endfortrue
7100     }%
7101     {}%
7102   }%
7103   \ifcsdef{gls\@gls@thisval accessdisplay}%
7104   {\letcs\@glxtr@accessdisplay{gls\@gls@thisval accessdisplay}}%
7105   {\let\@glxtr@accessdisplay\@firstoftwo}%
7106   }%
7107 }
7108 }
7109 {%
7110 \newcommand*{\glxtr@setaccessdisplay}[1]{%
7111   \let\@glxtr@accessdisplay\@firstoftwo
7112 }

```

`sentrynameother` Provide a command that works like `\glossentryname` but accesses a different field (which must be supplied using its internal field label).

```

7113 \newrobustcmd*{\glossentrynameother}[2]{%
7114   \@glsdoifexistsorwarn{#1}%
7115   {%

```

Accessibility support:

```

7116   \glxtr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

7117   \glsetabbrvfmt{\glscategory{#1}}%
7118   \glshasattribute{#1}{glossnamefont}%
7119   {%
7120     \protected@edef\@glxtr@attrval{\glsetattribute{#1}{glossnamefont}}%
7121     \ifcsdef{\@glxtr@attrval}%
7122     {%
7123       \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
7124     }%
7125     {%
7126       \GlossariesExtraWarning{Unknown control sequence name
7127         '\@glxtr@attrval' supplied in glossnamefont attribute

```

```

7128         for entry '#1'. Reverting to default \string\glsnamefont}%
7129         \let\@glsxtr@glossnamefont\glsnamefont
7130     }%
7131 }%
7132 {\let\@glsxtr@glossnamefont\glsnamefont}%
7133 \glsifattribute{#1}{glossname}{firstuc}%
7134 {%
7135     \@glsxtr@accessdisplay
7136     {\@glsxtr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
7137     {#1}%
7138 }%
7139 {%
7140     \glsifattribute{#1}{glossname}{title}%
7141     {%
7142         \@glsxtr@do@titlecaps@warn
7143         \@glsxtr@accessdisplay
7144         {\@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{#2}}}%
7145         {#1}%
7146     }%
7147     {%
7148         \glsifattribute{#1}{glossname}{uc}%
7149         {%
7150             \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
7151             \@glsxtr@accessdisplay
7152             {\@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
7153             {#1}%
7154         }%
7155         {%
7156             \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
7157             \@glsxtr@accessdisplay
7158             {\expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}}%
7159             {#1}%
7160         }%
7161     }%
7162 }%

```

Do post-name hook.

```

7163     \glsxtrpostnamehook{#1}%
7164 }%
7165 }

```

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

```

7166 \newif\if@glsxtr@format@override
7167 \@glsxtr@format@overridefalse

```

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

`xFormatOverride`

```

7168 \@ifpackageloaded{hyperref}

```



```

7169 {
    If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
    don't add it.
7170   \ifHy@hyperindex
7171     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
7172       \@glsxtr@format@overridetrue
7173       \appto\theindex{\let\glshypernumber\@firstofone}%
7174     }
7175   \else
7176     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
7177       \@glsxtr@format@overridetrue
7178       \appto\theindex{\let\glshypernumber\hyperpage}%
7179     }
7180   \fi
7181 }
7182 {
7183   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
7184     \@glsxtr@format@overridetrue
7185   }
7186 }
7187 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

doautoindexname

```

7188 \newcommand*{\glxtrdoautoindexname}[2]{%
7189   \glshasattribute{#1}{#2}%
7190   {%
    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.
7191     \@glsxtr@autoindex@setname{#1}%
    If the attribute value is simply "true" don't add an encap, otherwise use the value as the encap.
7192     \protected@edef\@glsxtr@attrval{\glsggetattribute{#1}{#2}}%
7193     \if@glsxtr@format@override
7194       \ifx\@glsnnumberformat\@glsxtr@defaultnumberformat
7195       \else
7196         \let\@glsxtr@attrval\@glsnnumberformat
7197       \fi
7198     \fi
7199     \ifdefstring{\@glsxtr@attrval}{true}%
7200     {}%
7201     {\protected@eappto\@glo@name{\@glsxtr@autoindex@encap\@glsxtr@attrval}}%
7202     \expandafter\glxtrautoindex\expandafter{\@glo@name}%
7203   }%
7204   {}%
7205 }

```

glsxtrautoindex

```
7206 \newcommand*{\glsxtrautoindex}{\index}
```

xtrautoindexesc

```
7207 \newcommand{\glsxtrautoindexesc}{%  
7208   \@gls@checkmkidxchars\@glo@sort  
7209   \@glsxtr@autoindex@doextra@esc\@glo@sort  
7210 }
```

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

```
7211 \newcommand*{\@glsxtr@autoindex@setname}[1]{%  
7212   \protected@edef\@glo@name{\glsxtrautoindexentry{#1}}%  
7213   \glsxtrautoindexassignsort{\@glo@sort}{#1}%  
7214   \glsxtrautoindexesc  
7215   \epreto\@glo@name{\@glo@sort\@glsxtr@autoindex@at}%  
7216 }
```

rautoindexentry Command used for the actual part when auto-indexing.

```
7217 \newcommand*{\glsxtrautoindexentry}[1]{\string\glsentryname{#1}}
```

indexassignsort Used to assign the sort value when auto-indexing.

```
7218 \newcommand*{\glsxtrautoindexassignsort}[2]{%  
7219   \glsletentryfield{#1}{#2}{sort}%  
7220 }
```

dex@doextra@esc

```
7221 \newcommand*{\@glsxtr@autoindex@doextra@esc}[1]{%  
    Escape the escape character unless it has already been escaped.  
7222   \ifx\@glsxtr@autoindex@esc\@gls@quotechar  
7223   \else  
7224     \def\@gls@checkedmkidx{%  
7225       \edef\@glsxtr@checkspch{%  
7226         \noexpand\@glsxtr@autoindex@escquote\expandonce{#1}%  
7227         \noexpand\@empty\@glsxtr@autoindex@esc\noexpand\@nnil  
7228         \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%  
7229         \@glsxtr@checkspch  
7230         \let#1\@gls@checkedmkidx\relax  
7231       \fi
```

Escape actual character unless it has already been escaped.

```
7232   \ifx\@glsxtr@autoindex@at\@gls@actualchar  
7233   \else  
7234     \def\@gls@checkedmkidx{%  
7235       \edef\@glsxtr@checkspch{%  
7236         \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%  
7237         \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil  
7238         \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%  
7239         \@glsxtr@checkspch
```

```
7240 \let#1\@gls@checkedmkidx\relax
```

```
7241 \fi
```

Escape level character unless it has already been escaped.

```
7242 \ifx\@glsxtr@autoindex@level\@gls@levelchar
```

```
7243 \else
```

```
7244 \def\@gls@checkedmkidx{}%
```

```
7245 \edef\@glsxtr@checkspch{%
```

```
7246 \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
```

```
7247 \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
```

```
7248 \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
```

```
7249 \@glsxtr@checkspch
```

```
7250 \let#1\@gls@checkedmkidx\relax
```

```
7251 \fi
```

Escape encap character unless it has already been escaped.

```
7252 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
```

```
7253 \else
```

```
7254 \def\@gls@checkedmkidx{}%
```

```
7255 \edef\@glsxtr@checkspch{%
```

```
7256 \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
```

```
7257 \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
```

```
7258 \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
```

```
7259 \@glsxtr@checkspch
```

```
7260 \let#1\@gls@checkedmkidx\relax
```

```
7261 \fi
```

```
7262 }
```

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.

```
7263 \newcommand*{\@glsxtr@autoindex@at}{}
```

trSetActualChar Set the actual character.

```
7264 \newcommand*{\GlsXtrSetActualChar}[1]{%
```

```
7265 \gdef\@glsxtr@autoindex@at{#1}%
```

```
7266 \def\@glsxtr@autoindex@escat##1#1##2#1##3\@glsxtr@endescspch{%
```

```
7267 \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escat}{##1}{##2}{##3}%
```

```
7268 }%
```

```
7269 }
```

```
7270 \@onlypreamble\GlsXtrSetActualChar
```

```
7271 \makeatother
```

```
7272 \GlsXtrSetActualChar{@}
```

```
7273 \makeatletter
```

autoindex@encap Encap character for use with \index.

```
7274 \newcommand*{\@glsxtr@autoindex@encap}{}
```

XtrSetEncapChar Set the encap character.

```
7275 \newcommand*{\GlsXtrSetEncapChar}[1]{%
7276   \gdef\@glstr@autoindex@encap{#1}%
7277   \def\@glstr@autoindex@escencap##1##2##3\@glstr@endescspch{%
7278     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escencap}{##1}{##2}{##3}%
7279   }%
7280 }
7281 \GlsXtrSetEncapChar{||}
7282 \@onlypreamble\GlsXtrSetEncapChar
```

autoindex@level Level character for use with \index.

```
7283 \newcommand*{\@glstr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
7284 \newcommand*{\GlsXtrSetLevelChar}[1]{%
7285   \gdef\@glstr@autoindex@level{#1}%
7286   \def\@glstr@autoindex@esclevel##1##2##3\@glstr@endescspch{%
7287     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@esclevel}{##1}{##2}{##3}%
7288   }%
7289 }
7290 \GlsXtrSetLevelChar{!}
7291 \@onlypreamble\GlsXtrSetLevelChar
```

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

```
7292 \newcommand*{\@glstr@autoindex@esc}{"}
```

lsXtrSetEscChar Set the escape character.

```
7293 \newcommand*{\GlsXtrSetEscChar}[1]{%
7294   \gdef\@glstr@autoindex@esc{#1}%
7295   \def\@glstr@autoindex@escquote##1##2##3\@glstr@endescspch{%
7296     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escquote}{##1}{##2}{##3}%
7297   }%
7298 }
7299 \GlsXtrSetEscChar{"}
7300 \@onlypreamble\GlsXtrSetEscChar
```

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

```
7301 \ifdef\actualchar
7302 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
7303 {}
```

Quote character \quotechar:

```
7304 \ifdef\quotechar
7305 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
7306 {}
```

Level character \levelchar:

```
7307 \ifdef\levelchar
7308 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
7309 {}
```

Encap character \encapchar:

```
7310 \ifdef\encapchar
7311 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
7312 {}}
```

leto@endescspch

```
7313 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

oindex@esc@spch

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

```
7314 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
7315   \@glstmpb=\expandafter{\@glstmpb\@glstmpb}%
7316   \toks@={#3}%
7317   \ifx\@nnil#3\relax
7318     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
7319   \else
7320     \ifx\@nnil#4\relax
7321       \edef\@glstmpb{\@glstmpb\@glstmpb}%
7322       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
7323         #4#5\@glxtr@endescspch}%
7324     \else
7325       \edef\@glstmpb{\@glstmpb\@glstmpb}%
7326       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
7327         #4#5\@glxtr@endescspch}%
7328     \fi
7329   \fi
7330   \@glxtr@checkspch
7331 }
```

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

```
7332 \renewcommand*{\Glossentrydesc}[1]{%
7333   \glstmpb=\@glstmpb\@glstmpb%
7334   {%
7335     \glstmpb=\@glstmpb\@glstmpb%
7336     \glstmpb=\@glstmpb\@glstmpb%
7337   }%
7338 }
```

lossentrysymbol Redefine to set the format and accessibility support. Allow for the possibility of being used in a section heading for standalone entry definitions.

```
7339 \ifdef\texorpdfstring
7340 {
7341   \renewcommand*{\lossentrysymbol}[1]{%
7342     \texorpdfstring{\@glstmpb\@glstmpb}{\@glstmpb\@glstmpb}%
7343   }
7344 }
```

```

7344 }
7345 {
7346   \renewcommand*{\glossentrysymbol}[1]{\@glossentrysymbol{#1}}
7347 }

```

`\glossentrypdfsymbol` May be redefined to a field that expands to a value that's more suitable for PDF bookmarks.

```

7348 \newcommand{\glsentrypdfsymbol}[1]{\glsentrysymbol{#1}}

```

`\glossentrysymbol` There are no case-changing attributes as it's less usual for symbols.

```

7349 \newrobustcmd*{\@glossentrysymbol}[1]{%
7350   \glsdoifexistsorwarn{#1}%
7351   {%
7352     \begingroup
7353     \glssetabbrvfmt{\glscategory{#1}}%
7354     \glshasattribute{#1}{glosssymbolfont}%
7355     {%
7356       \protected@edef\@glxtr@attrval{\glsggetattribute{#1}{glosssymbolfont}}%
7357       \ifcsdef{\@glxtr@attrval}%
7358       {%
7359         \letcs{\@glxtr@glosssymbolfont}{\@glxtr@attrval}%
7360       }%
7361       {%
7362         \GlossariesExtraWarning{Unknown control sequence name
7363           '\@glxtr@attrval' supplied in glosssymbolfont attribute
7364           for entry '#1'. Ignoring}%
7365         \let\@glxtr@glosssymbolfont\@firstofone
7366       }%
7367     }%
7368     {\let\@glxtr@glosssymbolfont\@firstofone}%
7369     \@glxtr@glosssymbolfont{\glsaccesssymbol{#1}}%
7370   }%
7371 }%
7372 }

```

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

```

7373 \renewcommand*{\Glossentrysymbol}[1]{%
7374   \glsdoifexistsorwarn{#1}%
7375   {%
7376     \glssetabbrvfmt{\glscategory{#1}}%
7377     \Glsaccesssymbol{#1}%
7378   }%
7379 }

```

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.

```

7380 \newcommand*{\GlsXtrEnableInitialTagging}{%

```

```

7381 \@ifstar\s@glxtr@enabletagging\@glxtr@enabletagging
7382 }
7383 \@onlypreamble\GlsXtrEnableInitialTagging

```

r@enabletagging Starred version undefines command.

```

7384 \newcommand*{\s@glxtr@enabletagging}[2]{%
7385   \undef#2%
7386   \@glxtr@enabletagging{#1}{#2}%
7387 }

```

r@enabletagging Internal command.

```

7388 \newcommand*{\@glxtr@enabletagging}[2]{%
    Set attributes for categories given in the first argument.
7389   \@for\@glxtr@cat:=#1\do
7390   {%
7391     \ifdefempty\@glxtr@cat
7392     {}%
7393     {\glsssetcategoryattribute{\@glxtr@cat}{tagging}{true}}%
7394   }%
7395   \newrobustcmd*#2[1]{##1}%
7396   \def\@glxtr@taggingcs{#2}%
7397   \renewcommand*\@glxtr@activate@initialtagging{%
7398     \let#2\@glxtr@tag
7399   }%
7400   \ifundef\@gl@preglossaryhook
7401   {\GlossariesExtraWarning{Initial tagging requires at least
7402     glossaries.sty v4.19 to work correctly}}%
7403   {}%
7404 }

```

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

```

7405 \ifundef\mfu@checkword@do
7406 {
7407   \newcommand*{\mfu@checkword@do}[1]{%
7408     \ifdefstring{\mfu@checkword@arg}{#1}%
7409     {%
7410       \let\@mfu@domakefirstuc\@firstofone
7411       \listbreak
7412     }%
7413     {}%
7414   }

```

\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.

```

7415 \ifundef\mfu@checkword

```

```

7416 {
7417   \newcommand{\@glxtr@do@titlecaps@warn}{%
7418     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
7419       support not available}%
       One warning should suffice.
7420     \let\@glxtr@do@titlecaps@warn\relax
7421   }
7422 }
7423 {
7424   \renewcommand*\mfu@checkword}[1]{%
7425     \def\mfu@checkword@arg{#1}%
7426     \let\@mfu@domakefirstuc\makefirstuc
7427     \forlistloop\mfu@checkword@do\@mfu@nocaplist
7428   }
7429 }
7430 }
7431 {}% no patch required

```

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

```
7432 \newcommand*\@glxtr@do@titlecaps@warn}{}
```

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

```
7433 \newcommand*\@glxtr@activate@initialtagging{}
```

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

```

7434 \newrobustcmd*\@glxtr@tag}[1]{%
7435   \gl@ifattribute{\gl@currententrylabel}{tagging}{true}%
7436   {\glxtr@tagfont{#1}}{#1}%
7437 }

```

\glxtr@tagfont Used in the glossary.

```
7438 \newcommand*\glxtr@tagfont}[1]{\underline{#1}}
```

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

```

7439 \ifdef\@gl@preglossaryhook
7440 {
7441   \renewcommand*\@gl@preglossaryhook}{%
7442     \@glxtr@activate@initialtagging

```

Since the glossaries are automatically scoped, \@glxtr@org@postdescription shouldn't already be defined, but check anyway just as a precautionary measure.

```

7443   \ifundef\@glxtr@org@postdescription
7444   {%
7445     \let\@glxtr@org@postdescription\glspostdescription
7446     \renewcommand*\glspostdescription}{%

```



```

7447     \ifglentryexists{\glscurrententrylabel}%
7448     {%
7449     \glxtrpostdescription
7450     \@glxtr@org@postdescription
7451     }%
7452     {}%
7453     }%
7454     }%
7455     {}%

```

Enable the options used by \@glxtrp:

```

7456     \glossxtrsetpopts
7457     }%
7458 }
7459 {}

```

`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.

```

7460 \newcommand*{\glxtrpostdescription}{%
7461   \csuse{glxtrpostdesc\glscategory{\glscurrententrylabel}}%
7462 }

```

`postdescgeneral`

```

7463 \newcommand*{\glxtrpostdescgeneral}{}

```

`xtrpostdescterm`

```

7464 \newcommand*{\glxtrpostdescterm}{}

```

`postdescacronym`

```

7465 \newcommand*{\glxtrpostdescacronym}{}

```

`escabbreviation`

```

7466 \newcommand*{\glxtrpostdescabbreviation}{}

```

`\glsdefpostdesc` Provide a convenient command for defining the post-description hook for the given category.

```

7467 \newcommand*{\glsdefpostdesc}[2]{%
7468   \csdef{glxtrpostdesc#1}{#2}%
7469 }

```

`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.

```

7470 \renewcommand*{\glspostlinkhook}{%
7471   \ifglentryexists{\glslabel}{\glxtrpostlinkhook}{}%
7472 }

```

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

```
7473 \newcommand*{\glsxtrpostlinkhook}{%
7474   \glsxtrdiscardperiod{\glslabel}%
7475   {\glsxtrpostlinkendsentence}%
7476   {\glsxtrifcustomdiscardperiod
7477     {\glsxtrifperiod{\glsxtrpostlinkendsentence}{\glsxtrpostlink}}}%
7478   {\glsxtrpostlink}%
7479 }%
7480 }
```

omdiscardperiod Allow user to provide a custom check. Should expand to #2 if no check is required otherwise expand to #1.

```
7481 \newcommand*{\glsxtrifcustomdiscardperiod}[2]{#2}
```

\glsxtrpostlink

```
7482 \newcommand*{\glsxtrpostlink}{%
7483   \csuse{\glsxtrpostlink\glscategory{\glslabel}}%
7484 }
```

\glsdefpostlink Provide a convenient command for defining the post-link hook for the given category. Doesn't allow an empty argument (which) would overwrite \glsxtrpostlink.

```
7485 \newcommand*{\glsdefpostlink}[2]{%
    \ifthenelse is used to ensure that the expanded value is tested. (The category label must
    be fully expandable.)
7486   \ifthenelse{\equal{#1}{}}{%
7487     {\PackageError{glossaries-extra}
7488       {Invalid empty category label in \string\glsdefpostlink}{}}%
7489     {\csdef{\glsxtrpostlink#1}{#2}}%
7490 }
```

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

```
7491 \newcommand*{\glsxtrpostlinkendsentence}{%
7492   \ifcsdef{\glsxtrpostlink\glscategory{\glslabel}}
7493   {%
7494     \csuse{\glsxtrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
7495     .\spacefactor\sfcode'\. \relax
7496   }%
7497   {%
```

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

```
7498     \spacefactor\sfcode'\. \relax
7499   }%
7500 }
```

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.

```
7501 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
7502   \glxtrifwasfirstuse{\space\glxtrparen{\glssaccessdesc{\glslabel}}}%
7503 }
```

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.

```
7504 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
7505   \glxtrifwasfirstuse
7506   {%
7507     \ifglshassymbol{\glslabel}%
7508     {\space\glxtrparen{\glssaccesssymbol{\glslabel}}}%
7509     }%
7510   }%
7511   {}%
7512 }
```

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

```
7513 \newcommand*{\glxtrpostlinkAddSymbolDescOnFirstUse}{%
7514   \glxtrifwasfirstuse
7515   {%
7516     \space\glxtrparen
7517     {%
7518       \ifglshassymbol{\glslabel}%
7519       {\glssaccesssymbol{\glslabel}, }%
7520       }%
7521       \glssaccessdesc{\glslabel}%
7522     }%
7523   }%
7524   {}%
7525 }
```

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).

```
7526 \newcommand*{\glxtrdiscardperiod}[3]{%
7527   \glxtrifwasfirstuse
7528   {%
7529     \glusifattribute{#1}{retainfirstuseperiod}{true}%
7530     {#3}%
7531     {%
7532       \glusifattribute{#1}{discardperiod}{true}%
7533       {%
7534         \glusifplural
7535         {%
```

```

7536     \glsifattribute{#1}{pluraldiscardperiod}{true}%
7537     {\glsxtrifperiod{#2}{#3}}%
7538     {#3}%
7539     }%
7540     {%
7541     \glsxtrifperiod{#2}{#3}%
7542     }%
7543     }%
7544     {#3}%
7545     }%
7546 }%
7547 {%
7548 \glsifattribute{#1}{discardperiod}{true}%
7549 {%
7550 \glsifplural
7551 {%
7552 \glsifattribute{#1}{pluraldiscardperiod}{true}%
7553 {\glsxtrifperiod{#2}{#3}}%
7554 {#3}%
7555 }%
7556 {%
7557 \glsxtrifperiod{#2}{#3}%
7558 }%
7559 }%
7560 {#3}%
7561 }%
7562 }

```

`\glsxtrifperiod` 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`

```
7563 \newcommand*{\glsxtrifperiod}[1]{\new@ifnextchar.{\@firstoftwo{#1}}}
```

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

`\glsxtr@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 ' ').

```
7564 \newcommand*{\glsxtr@punclist}{.,:;?!}
```

`\punctuationmark` Add character to punctuation list.

```
7565 \newcommand*{\glsxtraddpunctuationmark}[1]{\appto\glsxtr@punclist{#1}}
```

`\punctuationmarks` Reset the punctuation list.

```
7566 \newcommand*{\glsxtrsetpunctuationmarks}[1]{\def\glsxtr@punclist{#1}}
```

`\glsxtrifpunc`

`\glsxtrifnextpunc{<true part>}{<false part>}`

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

```

7567 \newcommand*{\glxtrifnextpunc}[2]{%
7568   \def\reserved@a{#1}%
7569   \def\reserved@b{#2}%
7570   \futurelet\@glspunc@token\glxtr@ifnextpunc
7571 }

```

`glxtr@ifnextpunc`

```

7572 \newcommand*{\glxtr@ifnextpunc}{%
7573   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}}%
7574   \reserved@b
7575 }

```

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

```

7576 \newcommand*{\glxtr@ifpunctoken}[1]{%
7577   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
7578 }

```

`glxtr@ifpunctoken`

```

7579 \def\@glxtr@ifpunctoken#1#2{%
7580   \let\reserved@d=#2%
7581   \ifx\reserved@d\@nnil
7582     \let\glxtr@next\@glxtr@notfoundinlist
7583   \else
7584     \ifx#1\reserved@d
7585       \let\glxtr@next\@glxtr@foundinlist
7586     \else
7587       \let\glxtr@next\@glxtr@ifpunctoken
7588     \fi
7589   \fi
7590   \glxtr@next#1%
7591 }

```

`glxtr@foundinlist`

```

7592 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}

```

`@notfoundinlist`

```

7593 \def\@glxtr@notfoundinlist#1{\@secondoftwo}

```

`glxtrdopostpunc`

`\glxtrdopostpunc{<code>}`

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

```

7594 \newcommand{\glxtrdopostpunc}[1]{%
7595   \glxtrifnextpunc{\@glxtr@swaptwo{#1}}{#1}%
7596 }

```

@glxstr@swaptwo

```
7597 \newcommand{\@glxstr@swaptwo}[2]{#2#1}
```

1.7 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, it needs to be applied by \newabbreviation.

```
7598 \define@key{glxstrabbrv}{category}{%
7599   \protected@edef\glscategorylabel{#1}%
7600 }
```

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

```
7601 \define@key{glxstrabbrv}{shortplural}{%
7602   \def\@gls@shortpl{#1}%
7603 }
```

Similarly for the long plural form.

```
7604 \define@key{glxstrabbrv}{longplural}{%
7605   \def\@gls@longpl{#1}%
7606 }
```

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

\glsshortpltok

```
7607 \newtoks\glsshortpltok
```

\glslongpltok

```
7608 \newtoks\glslongpltok
```

glxstr@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.

```
7609 \newcommand*{\@glxstr@insertdots}[2]{%
7610   \def#1{%
7611     \@glxstr@insert@dots#1#2\@nnil
7612   }}
```

glxstr@insert@dots

```
7613 \newcommand*{\@glxstr@insert@dots}[2]{%
7614   \ifx\@nnil#2\relax
```

```

7615 \let\@glsxtr@insert@dots@next\@gobble
7616 \else
7617 \ifx\relax#2\relax
7618 \else
7619 \appto#1{#2.}%
7620 \fi
7621 \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
7622 \fi
7623 \@glsxtr@insert@dots@next#1%
7624 }

```

Similarly provide a way of replacing spaces with `\glsxtrwordsep`, which first needs to be defined:

`\glsxtrwordsep`

```

7625 \newcommand*{\glsxtrwordsep}{\space}

```

Each word is marked with

`\glsxtrword`

```

7626 \newcommand*{\glsxtrword}[1]{#1}

```

`\@glsxtr@mark@wordseps`

```

7627 \newcommand*{\@glsxtr@mark@wordseps}[2]{%
7628 \def#1{}%
7629 \@glsxtr@mark@wordseps#1#2 \@nnil
7630 }

```

`\@glsxtr@mark@wordseps`

```

7631 \def\@glsxtr@mark@wordseps#1#2 #3{%
7632 \ifdefempty{#1}%
7633 {\def#1{\protect\glsxtrword{#2}}}%
7634 {\appto#1{\protect\glsxtrwordsep\protect\glsxtrword{#2}}}%
7635 \ifx\@nnil#3\relax
7636 \let\@glsxtr@mark@wordseps@next\relax
7637 \else
7638 \def\@glsxtr@mark@wordseps@next{%
7639 \@glsxtr@mark@wordseps#1#3}%
7640 \fi
7641 \@glsxtr@mark@wordseps@next
7642 }

```

`\newabbreviation` Define a new generic abbreviation.

```

7643 \newcommand*{\newabbreviation}[4][ ]{%
7644 \glsxtr@newabbreviation{#1}{#2}{#3}{#4}%
7645 }

```

`\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.)

```

7646 \newcommand*{\glxtr@newabbreviation}[4]{%
7647   \glskeylisttok{#1}%
7648   \glslabeltok{#2}%
7649   \glsshorttok{#3}%
7650   \glslongtok{#4}%

```

Save the original short and long values (before attribute settings modify them).

```

7651   \def\glxtrorgshort{#3}%
7652   \def\glxtrorglong{#4}%

```

Provide extra settings for hooks (if modified, this command must end with a comma).

```

7653   \def\ExtraCustomAbbreviationFields{}%

```

Initialise accessibility settings if required.

```

7654   \@gls@initaccesskeys

```

Get the category.

```

7655   \def\glscategorylabel{abbreviation}%

```

Ignore the shortplural and longplural keys.

```

7656   \setkeys*{\glxtrabbrv}[shortplural,longplural]{#1}%

```

Set the abbreviation style.

```

7657   \ifcsdef{@glsabbrv@current@\glscategorylabel}%
7658   {%

```

Warning should already have been issued.

```

7659     \let\glxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
7660     \let\GlsXtrWarnDeprecatedAbbrStyle@gobbletwo
7661     \glxtr@applyabbrvstyle{\csname @glsabbrv@current@\glscategorylabel\endcsname}%
7662     \let\GlsXtrWarnDeprecatedAbbrStyle\glxtr@orgwarndep
7663   }%
7664   {%

```

If no style has been associated with this category, fallback on the style for the abbreviation category.

```

7665     \glxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
7666   }%

```

Set the default long plural

```

7667   \def\@gls@longpl{#4\glspluralsuffix}%
7668   \let\@gls@default@longpl\@gls@longpl

```

Has the markwords attribute been set?

```

7669   \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
7670   {%
7671     \@glxtr@keywordseps\@gls@long{#4}%
7672     \expandafter\def\expandafter\@gls@longpl\expandafter
7673       {\@gls@long\glspluralsuffix}%
7674     \let\@gls@default@longpl\@gls@longpl

```

Update \glslongtok.

```

7675     \expandafter\glslongtok\expandafter{\@gls@long}%
7676   }%
7677   {}%

```


Has the markshortwords attribute been set? (Not compatible with insertdots.)

```
7678 \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%  
7679 {%  
7680 \@glstr@markwordseps\@gls@short{#3}%  
7681 }%  
7682 {%
```

Has the insertdots attribute been set?

```
7683 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%  
7684 {%  
7685 \@glstr@insertdots\@gls@short{#3}%  
  
7686 \appto\@gls@short{\@}%  
7687 }%  
7688 {\def\@gls@short{#3}}%  
7689 }%
```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```
7690 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%  
7691 {%  
7692 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
7693 '\abrvpluralsuffix}%  
7694 }%  
7695 {%
```

Has the noshortplural attribute been set?

```
7696 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%  
7697 {%  
7698 \let\@gls@shortpl\@gls@short  
7699 }%  
7700 {%  
7701 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
7702 \abrvpluralsuffix}%  
7703 }%  
7704 }%
```

Update \glsshorttok:

```
7705 \expandafter\glsshorttok\expandafter{\@gls@short}%
```

Hook for further customisation if required:

```
7706 \glstrnewabbrevpresetkeyhook{#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).

```
7707 \setkeys*{glstrabbrv}[category]{#1}%
```

Save in case required.

```
7708 \let\@gls@org@longpl\@gls@longpl  
7709 \let\@gls@org@shortpl\@gls@shortpl
```

Has the plural been explicitly set?

```
7710 \ifx\@gls@default@longpl\@gls@longpl  
7711 \else
```

Has the markwords attribute been set?

```
7712 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
7713 {%
7714 \expandafter\@glstr@markwordseps\expandafter\@gls@longpl\expandafter
7715 {\@gls@longpl}%
7716 }%
7717 {}%
7718 \fi
```

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

```
7719 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
7720 \expandafter\glslongpltok\expandafter{\@gls@longpl}%
```

Hook for accessibility support (does nothing if glossaries-accsupp hasn't been loaded).

```
7721 \@gls@setup@default@access
```

Do any extra setup provided by hook:

```
7722 \newabbreviationhook
```

Define this entry:

```
7723 \protected@edef\@do@newglossaryentry{%
7724 \noexpand\newglossaryentry{\the\glslabeltok}%
7725 {%
7726 type=\glstrabbrvtype,%
7727 category=abbreviation,%
7728 short={\the\glsshorttok},%
7729 shortplural={\the\glsshortpltok},%
7730 long={\the\glslongtok},%
7731 longplural={\the\glslongpltok},%
7732 name={\the\glsshorttok},%
7733 \CustomAbbreviationFields,%
```

Hook may override abbreviation style default settings (this hook must end with a comma if set).

```
7734 \ExtraCustomAbbreviationFields
```

Any explicit fields set in the optional argument override all other settings.

```
7735 \the\glskeylisttok
7736 }%
7737 }%
7738 \@do@newglossaryentry
```

Obtain the type and add it to the list of abbreviations.

```
7739 \@glstr@addabbreviationlist{\glstentrytype{\the\glslabeltok}}%
7740 \GlsXtrPostNewAbbreviation
7741 }
```

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

```
7742 \newcommand*{\glstrnewabbrevpresetkeyhook}[3]{}%
```

`\NewAbbreviation` Hook used by abbreviation styles.

```
7743 \newcommand*{\GlsXtrPostNewAbbreviation}{}%
```

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

```
7744 \newcommand*{\newabbreviationhook}{}

```

`\CustomAbbreviationFields`

```
7745 \newcommand*{\CustomAbbreviationFields}{}

```

`\glstrparen` For the parenthetical styles.

```
7746 \newcommand*{\glstrparen}[1]{(#1)}

```

`\glstrfullformat` Full format without case change.

```
7747 \newcommand*{\glstrfullformat}[2]{%
7748   \glsfirstlongfont{\glssaccesslong{#1}}#2\glstrfullsep{#1}%
7749   \glstrparen{\protect\glsfirstabbrvfont{\glssaccessshort{#1}}}%
7750 }

```

`\Glsstrfullformat` Full format with case change.

```
7751 \newcommand*{\Glsstrfullformat}[2]{%
7752   \glsfirstlongfont{\Glsaccesslong{#1}}#2\glstrfullsep{#1}%
7753   \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
7754 }

```

`\glstrfullplformat` Plural full format without case change.

```
7755 \newcommand*{\glstrfullplformat}[2]{%
7756   \glsfirstlongfont{\glssaccesslongpl{#1}}#2\glstrfullsep{#1}%
7757   \glstrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{#1}}}%
7758 }

```

`\Glsstrfullplformat` Plural full format with case change.

```
7759 \newcommand*{\Glsstrfullplformat}[2]{%
7760   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glstrfullsep{#1}%
7761   \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
7762 }

```

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

```
7763 \newcommand*{\glstrfullsep}[1]{\space}

```

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

`\glstrinlinefullformat` Full format without case change.

```
7764 \newcommand*{\glstrinlinefullformat}{\glstrfullformat}

```

`\Glsstrinlinefullformat` Full format with case change.

```
7765 \newcommand*{\Glsstrinlinefullformat}{\Glsstrfullformat}

```

`\glstrfullplformat` Plural full format without case change.

```
7766 \newcommand*{\glstrinlinefullplformat}{\glstrfullplformat}

```

`\inlinefullplformat` Plural full format with case change.
7767 `\newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}`

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

`\glsentryfull`
7768 `\renewcommand*{\glsentryfull}[1]{\glsxtrinlinefullformat{#1}{}}`

`\Glsentryfull`
7769 `\renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}`

`\glsentryfullpl`
7770 `\renewcommand*{\glsentryfullpl}[1]{\glsxtrinlinefullplformat{#1}{}}`

`\Glsentryfullpl`
7771 `\renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}`

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.
7772 `\newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.
7773 `\newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvdefaultfont{#1}}`

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use. This is redefined by the abbreviation styles, as appropriate.
7774 `\newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont`
7775 `\newcommand*{\glsabbrvdefaultfont}[1]{#1}`

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.
7776 `\newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}`

`\glslongdefaultfont` Default font changing command used for the long form in commands like `\glsxtrlong`.
7777 `\newcommand*{\glslongdefaultfont}[1]{#1}`

`\glsfirstlongfont` Font changing command used for the long form on first use or in the full format.
7778 `\newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}`

`\glsfirstlongdefaultfont`
7779 `\newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}`

`\glsbrvpluralsuffix` Default plural suffix. Allow an alternative default suffix for abbreviations.
7780 `\newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}`

brvpluralsuffix Default plural suffix.

```
7781 \newcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}
```

\glxtrfull Full form (no case-change).

```
7782 \newrobustcmd*{\glxtrfull}{\@gls@hyp@opt\ns@glxtrfull}
7783 \newcommand*\ns@glxtrfull[2] [] {%
7784   \new@ifnextchar[{\@glxtr@full{#1}{#2}}{%
7785     {\@glxtr@full{#1}{#2} []}%
7786 }
```

\@glxtr@full Low-level macro:

```
7787 \def\@glxtr@full#1#2[#3] {%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7788   \@glxtr@record{#1}{#2}{glslink}%
7789   \glsoifexists{#2}%
7790   {%
7791     \glsetabbrvfmt{\glscategory{#2}}%
7792     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7793     \let\gl@ifplural\@secondoftwo
7794     \let\gl@scapscase\@firstofthree
7795     \let\gl@insert\@empty
7796     \def\glscustomtext{\glxtrinlinefullformat{#2}{#3}}%
```

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

```
7797   \glxtrsetupfulldefs
7798   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7799   }%
7800   \glspostlinkhook
7801 }
```

trsetupfulldefs

```
7802 \newcommand*{\glxtrsetupfulldefs}{%
7803   \let\glxtrifwasfirstuse\@firstoftwo
7804 }
```

\Glsxtrfull Full form (first letter uppercase).

```
7805 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\ns@Glsxtrfull}
7806 \newcommand*\ns@Glsxtrfull[2] [] {%
7807   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}{%
7808     {\@Glsxtr@full{#1}{#2} []}%
7809 }
```

\@Glsxtr@full Low-level macro:

```

7810 \def\@GLSxtr@full#1#2[#3]{%
7811   \glsdoifexists{#2}%
7812   {%
7813     \glsetabbrvfmt{\glscategory{#2}}%
7814     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7815     \let\gl@sifplural\@secondoftwo
7816     \let\glscapscase\@secondofthree
7817     \let\gl$insert\@empty
7818     \def\glscustomtext{\GLSxtrinlinefullformat{#2}{#3}}%
7819     \glxtrsetupfulldefs
7820     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7821   }%
7822   \glspostlinkhook
7823 }

```

\GLSxtrfull Full form (all uppercase).

```

7824 \newrobustcmd*{\GLSxtrfull}{\@gl@hyp@opt\@ns@GLSxtrfull}
7825 \newcommand*\ns@GLSxtrfull[2][{}]{%
7826   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}%
7827     {\@GLSxtr@full{#1}{#2}[]}%
7828 }

```

\@GLSxtr@full Low-level macro:

```

7829 \def\@GLSxtr@full#1#2[#3]{%
7830   \glsdoifexists{#2}%
7831   {%
7832     \glsetabbrvfmt{\glscategory{#2}}%
7833     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7834     \let\gl@sifplural\@secondoftwo
7835     \let\glscapscase\@thirdofthree
7836     \let\gl$insert\@empty
7837     \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
7838     \glxtrsetupfulldefs
7839     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7840   }%
7841   \glspostlinkhook
7842 }

```

\glxtrfullpl Plural full form (no case-change).

```

7843 \newrobustcmd*{\glxtrfullpl}{\@gl@hyp@opt\@ns@glxtrfullpl}
7844 \newcommand*\ns@glxtrfullpl[2][{}]{%
7845   \new@ifnextchar[{\@glxtr@fullpl{#1}{#2}}%
7846     {\@glxtr@fullpl{#1}{#2}[]}%
7847 }

```

\@glxtr@fullpl Low-level macro:

```

7848 \def\@glxtr@fullpl#1#2[#3]{%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

7849 \@glstr@record{#1}{#2}{glslink}%
7850 \glsoifexists{#2}%
7851 {%
7852   \glsetabbrvfmt{\glscategory{#2}}%
7853   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7854   \let\gl@ifplural\@firstoftwo
7855   \let\glscapscase\@firstofthree
7856   \let\glinsert\@empty
7857   \def\glscustomtext{\glstrinlinefullplformat{#2}{#3}}%
7858   \glstrsetupfulldefs
7859   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7860 }%
7861 \glspostlinkhook
7862 }

```

\Glsxtrfullpl Plural full form (first letter uppercase).

```

7863 \newrobustcmd*{\Glsxtrfullpl}{\@gl@hyp@opt\@ns@Glsxtrfullpl}
7864 \newcommand*\ns@Glsxtrfullpl[2][{}]{%
7865   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
7866   {\@Glsxtr@fullpl{#1}{#2}[]}%
7867 }

```

\@Glsxtr@fullpl Low-level macro:

```

7868 \def\@Glsxtr@fullpl#1#2[#3]{%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

7869 \@glstr@record{#1}{#2}{glslink}%
7870 \glsoifexists{#2}%
7871 {%
7872   \glsetabbrvfmt{\glscategory{#2}}%
7873   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7874   \let\gl@ifplural\@firstoftwo
7875   \let\glscapscase\@secondofthree
7876   \let\glinsert\@empty
7877   \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
7878   \glstrsetupfulldefs
7879   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7880 }%
7881 \glspostlinkhook
7882 }

```

\GLSxtrfullpl Plural full form (all upper case).

```

7883 \newrobustcmd*{\GLSxtrfullpl}{\@gl@hyp@opt\@ns@GLSxtrfullpl}
7884 \newcommand*\ns@GLSxtrfullpl[2][{}]{%
7885   \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}%
7886   {\@GLSxtr@fullpl{#1}{#2}[]}%

```

7887 }

\@GLSxtr@fullpl Low-level macro:

7888 \def\@GLSxtr@fullpl#1#2[#3]{%

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

7889 \@glxtr@record{#1}{#2}{glslink}%

7890 \glstoifexists{#2}%

7891 {%

7892 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper

7893 \let\gl@sifplural\@firstoftwo

7894 \let\glscapscase\@thirdofthree

7895 \let\gl\$insert\@empty

7896 \def\glscustomtext{%

7897 \mfirstucMakeUppercase{\glxtrinlinefullplformat{#2}{#3}}}%

7898 \glxtrsetupfulldefs

7899 \@gl@link[#1]{#2}{\csname gls\@glstype @entryfmt\endcsname}%

7900 }%

7901 \glspostlinkhook

7902 }

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

\glxtrshort

7903 \newrobustcmd*{\glxtrshort}{\@gl@hyp@opt\@ns@glxtrshort}

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

7904 \newcommand*{\@ns@glxtrshort}[2][]{%

7905 \new@ifnextchar[{\@glxtrshort{#1}{#2}}{\@glxtrshort{#1}{#2}[]}%

7906 }

Read in the final optional argument:

7907 \def\@glxtrshort#1#2[#3]{%

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

7908 \@glxtr@record{#1}{#2}{glslink}%

7909 \glstoifexists{#2}%

7910 {%

Need to make sure \gl@sabbrvfont is set correctly.

7911 \glsssetabbrvfmt{\glscategory{#2}}%

7912 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper

7913 \let\glxtrifwasfirstuse\@secondoftwo

7914 \let\gl@sifplural\@secondoftwo

7915 \let\glscapscase\@firstofthree

7916 \let\gl\$insert\@empty

7917 \def\glscustomtext{%

7918 \gl@sabbrvfont{\gl@saccessshort{#2}\ifglxtrininsertinside#3\fi}%

7919 \ifglxtrininsertinside\else#3\fi


```

7920 }%
7921 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7922 }%
7923 \glspostlinkhook
7924 }

```

\Glsxtrshort

```

7925 \newrobustcmd*{\Glsxtrshort}{\@gls@hyp@opt\ns@Glsxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
7926 \newcommand*{\ns@Glsxtrshort}[2][{}]{%
7927   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2}[]}%
7928 }

    Read in the final optional argument:
7929 \def\@Glsxtrshort#1#2[#3]{%

    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7930   \@glsxtr@record{#1}{#2}{glslink}%
7931   \glsdoifexists{#2}%
7932   {%
7933     \glssetabbrvfmt{\glscategory{#2}}%
7934     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7935     \let\glsxtrifwasfirstuse\@secondoftwo
7936     \let\glsifplural\@secondoftwo
7937     \let\glsapscase\@secondofthree
7938     \let\glsinsert\@empty
7939     \def\glscustomtext{%
7940       \glsabbrvfont{\Glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
7941       \ifglsxtrininsertinside\else#3\fi
7942     }%
7943     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7944   }%
7945   \glspostlinkhook
7946 }

```

\GLSxtrshort

```

7947 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\ns@GLSxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
7948 \newcommand*{\ns@GLSxtrshort}[2][{}]{%
7949   \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2}[]}%
7950 }

    Read in the final optional argument:
7951 \def\@GLSxtrshort#1#2[#3]{%

    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7952   \@glsxtr@record{#1}{#2}{glslink}%

```

```

7953 \glsdoifexists{#2}%
7954 {%
7955   \glssetabbrvfmt{\glscategory{#2}}%
7956   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7957   \let\glxtrifwasfirstuse\@secondoftwo
7958   \let\gl@ifplural\@secondoftwo
7959   \let\glscapscase\@thirdofthree
7960   \let\glinsert\@empty
7961   \def\glscustomtext{%
7962     \mfirstucMakeUppercase
7963     {\glabbrvfont{\glaccessshort{#2}\ifglxtrininsertinside#3\fi}%
7964     \ifglxtrininsertinside\else#3\fi
7965   }%
7966   }%
7967   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7968 }%
7969 \glspostlinkhook
7970 }

```

\glxtrlong

```

7971 \newrobustcmd*{\glxtrlong}{\@gl@hyp@opt\@ns@glxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
7972 \newcommand*{\ns@glxtrlong}[2][{}]{%
7973   \new@ifnextchar[{\@glxtrlong{#1}{#2}}{\@glxtrlong{#1}{#2}[]}%
7974 }

```

Read in the final optional argument:

```

7975 \def\@glxtrlong#1#2[#3]{%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7976   \@glxtr@record{#1}{#2}{glslink}%
7977   \glsdoifexists{#2}%
7978   {%
7979     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7980     \let\glxtrifwasfirstuse\@secondoftwo
7981     \let\gl@ifplural\@secondoftwo
7982     \let\glscapscase\@firstofthree
7983     \let\glinsert\@empty
7984     \def\glscustomtext{%
7985       \glslongfont{\glaccesslong{#2}\ifglxtrininsertinside#3\fi}%
7986       \ifglxtrininsertinside\else#3\fi
7987     }%
7988     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7989   }%
7990   \glspostlinkhook
7991 }

```

\Glsxtrlong

```

7992 \newrobustcmd*{\Glsxtrlong}{\@gl@hyp@opt\@ns@Glsxtrlong}

```

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

```
7993 \newcommand*{\ns@Glsxtrlong}[2] [] {%
7994   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
7995 }
```

Read in the final optional argument:

```
7996 \def\@Glsxtrlong#1#2[#3] {%
```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```
7997   \@glsxtr@record{#1}{#2}{glslink}%
7998   \glsdoifexists{#2}%
7999   {%
8000     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
8001     \let\glsxtrifwasfirstuse\@secondoftwo
8002     \let\glsifplural\@secondoftwo
8003     \let\gls caps case\@secondofthree
8004     \let\glsinsert\@empty
8005     \def\gls custom text{%
8006       \gls long font{\Gls access long{#2}\ifglsxtrininsertinside#3\fi}%
8007       \ifglsxtrininsertinside\else#3\fi
8008     }%
8009     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8010   }%
8011   \gls post link hook
8012 }
```

\GLSxtrlong

```
8013 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}
```

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

```
8014 \newcommand*{\ns@GLSxtrlong}[2] [] {%
8015   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
8016 }
```

Read in the final optional argument:

```
8017 \def\@GLSxtrlong#1#2[#3] {%
```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```
8018   \@glsxtr@record{#1}{#2}{glslink}%
8019   \glsdoifexists{#2}%
8020   {%
8021     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
8022     \let\glsxtrifwasfirstuse\@secondoftwo
8023     \let\glsifplural\@secondoftwo
8024     \let\gls caps case\@thirdofthree
8025     \let\glsinsert\@empty
8026     \def\gls custom text{%
8027       \mfirstucMakeUppercase
8028       {\gls long font{\Gls access long{#2}\ifglsxtrininsertinside#3\fi}%

```

```

8029      \ifglxtrinsertinside\else#3\fi
8030    }%
8031  }%
8032  \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8033 }%
8034 \glspostlinkhook
8035 }

```

Plural short forms:

\glxtrshortpl

```

8036 \newrobustcmd*{\glxtrshortpl}{\@gls@hyp@opt\ns@glxtrshortpl}

```

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

```

8037 \newcommand*{\ns@glxtrshortpl}[2][{}]{%
8038   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2}[]}%
8039 }

```

Read in the final optional argument:

```

8040 \def\@glxtrshortpl#1#2[#3]{%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

8041   \@glxtr@record{#1}{#2}{glslink}%
8042   \glstoifexists{#2}%
8043   {%
8044     \glsssetabbrvfmt{\glscategory{#2}}%
8045     \let\do@glis@link@checkfirsthyper\@glis@link@nocheckfirsthyper
8046     \let\glxtrifwasfirstuse\@secondoftwo
8047     \let\glisifplural\@firstoftwo
8048     \let\glscapscase\@firstofthree
8049     \let\glinsert\@empty
8050     \def\glscustomtext{%
8051       \glssabbrvfont{\glssaccessshortpl{#2}\ifglxtrinsertinside#3\fi}%
8052       \ifglxtrinsertinside\else#3\fi
8053     }%
8054     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8055   }%
8056   \glspostlinkhook
8057 }

```

\Glsxtrshortpl

```

8058 \newrobustcmd*{\Glsxtrshortpl}{\@gls@hyp@opt\ns@Glsxtrshortpl}

```

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

```

8059 \newcommand*{\ns@Glsxtrshortpl}[2][{}]{%
8060   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2}[]}%
8061 }

```

Read in the final optional argument:

```

8062 \def\@Glsxtrshortpl#1#2[#3]{%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

8063 \@glstr@record{#1}{#2}{glslink}%
8064 \glsoifexists{#2}%
8065 {%
8066 \glsetabbrvfmt{\glscategory{#2}}%
8067 \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
8068 \let\glstrifwasfirstuse\@secondoftwo
8069 \let\glslplural\@firstoftwo
8070 \let\glscapscase\@secondofthree
8071 \let\glslinsert\@empty
8072 \def\glscustomtext{%
8073 \glslabbrvfont{\Glsaccessshortpl{#2}\ifglstrinsertinside#3\fi}%
8074 \ifglstrinsertinside\else#3\fi
8075 }%
8076 \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8077 }%
8078 \glspostlinkhook
8079 }

```

\GLSxtrshortpl

```

8080 \newrobustcmd*{\GLSxtrshortpl}{\@glsl@hyp@opt\ns@GLSxtrshortpl}

```

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

```

8081 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
8082 \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
8083 }

```

Read in the final optional argument:

```

8084 \def\@GLSxtrshortpl#1#2[#3] {%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

8085 \@glstr@record{#1}{#2}{glslink}%
8086 \glsoifexists{#2}%
8087 {%
8088 \glsetabbrvfmt{\glscategory{#2}}%
8089 \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
8090 \let\glstrifwasfirstuse\@secondoftwo
8091 \let\glslplural\@firstoftwo
8092 \let\glscapscase\@thirdofthree
8093 \let\glslinsert\@empty
8094 \def\glscustomtext{%
8095 \mfirstucMakeUppercase
8096 {\glslabbrvfont{\Glsaccessshortpl{#2}\ifglstrinsertinside#3\fi}%
8097 \ifglstrinsertinside\else#3\fi
8098 }%
8099 }%
8100 \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8101 }%

```

```

8102 \glspostlinkhook
8103 }

```

Plural long forms:

\glsxtrlongpl

```

8104 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\ns@glxtrlongpl}
Define the un-starred form. Need to determine if there is a final optional argument
8105 \newcommand*{\ns@glxtrlongpl}[2] [] {%
8106 \new@ifnextchar[{\@glxtrlongpl{#1}{#2}}{\@glxtrlongpl{#1}{#2} []}%
8107 }

```

Read in the final optional argument:

```

8108 \def\@glxtrlongpl#1#2[#3] {%
If the record option has been used, the information needs to be written to the aux file regard-
less of whether the entry exists (unless indexing has been switched off).
8109 \@glxtr@record{#1}{#2}{glslink}%
8110 \glsdoifexists{#2}%
8111 {%
8112 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
8113 \let\glxtrifwasfirstuse\@secondoftwo
8114 \let\gl@ifplural\@firstoftwo
8115 \let\glscapscase\@firstofthree
8116 \let\glinsert\@empty
8117 \def\glscustomtext{%
8118 \glslongfont{\gl@saccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
8119 \ifglxtrininsertinside\else#3\fi
8120 }%
8121 \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
8122 }%
8123 \glspostlinkhook
8124 }

```

\Glsxtrlongpl

```

8125 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\ns@Glsxtrlongpl}
Define the un-starred form. Need to determine if there is a final optional argument
8126 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
8127 \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
8128 }

```

Read in the final optional argument:

```

8129 \def\@Glsxtrlongpl#1#2[#3] {%
If the record option has been used, the information needs to be written to the aux file regard-
less of whether the entry exists (unless indexing has been switched off).
8130 \@glxtr@record{#1}{#2}{glslink}%
8131 \glsdoifexists{#2}%
8132 {%

```

```

8133 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
8134 \let\glxtrifwasfirstuse\@secondoftwo
8135 \let\gl@sifplural\@firstoftwo
8136 \let\glscapscase\@secondofthree
8137 \let\gl$insert\@empty
8138 \def\glscustomtext{%
8139 \glslongfont{\Glsaccesslongpl{#2}\ifglxtrinsertinside#3\fi}%
8140 \ifglxtrinsertinside\else#3\fi
8141 }%
8142 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8143 }%
8144 \glspostlinkhook
8145 }

```

\GLSxtrlongpl

```

8146 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\@ns@GLSxtrlongpl}
      Define the un-starred form. Need to determine if there is a final optional argument
8147 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
8148 \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
8149 }

```

Read in the final optional argument:

```

8150 \def\@GLSxtrlongpl#1#2[#3]{%
      If the record option has been used, the information needs to be written to the aux file regard-
      less of whether the entry exists (unless indexing has been switched off).
8151 \@glxtr@record{#1}{#2}{\gl@link}%
8152 \gl@doifexists{#2}%
8153 {%
8154 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
8155 \let\glxtrifwasfirstuse\@secondoftwo
8156 \let\gl@sifplural\@firstoftwo
8157 \let\glscapscase\@thirdofthree
8158 \let\gl$insert\@empty
8159 \def\glscustomtext{%
8160 \mfirstucMakeUppercase
8161 {\glslongfont{\Glsaccesslongpl{#2}\ifglxtrinsertinside#3\fi}%
8162 \ifglxtrinsertinside\else#3\fi
8163 }%
8164 }%
8165 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8166 }%
8167 \glspostlinkhook
8168 }

```

\glsetabbrvfmt Set the current format for the given category (or the abbreviation category if unset).

```

8169 \newcommand*{\glsetabbrvfmt}[1]{%
8170 \ifcsdef{@gl@abbrv@current@#1}%
8171 {\glxtr@applyabbrvfmt{\csname @gl@abbrv@current@#1\endcsname}}%

```

```

8172 {\glxstr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
8173 }

```

glsuseabbrvfont Provide a way to use the abbreviation font for a given category for arbitrary text.

```

8174 \newrobustcmd*{\glsuseabbrvfont}[2]{\{\glsetabbrvfmt{#2}\glsabbrvfont{#1}}\}

```

\glsuselongfont Provide a way to use the long font for a given category for arbitrary text.

```

8175 \newrobustcmd*{\glsuselongfont}[2]{\{\glsetabbrvfmt{#2}\glslongfont{#1}}\}

```

sxtrgenabbrvfmt Similar to `\glsgenacfmt`, but for abbreviations.

```

8176 \newcommand*{\glsxtrgenabbrvfmt}{%
8177   \ifdefempty\glscustomtext
8178   {%
8179     \ifglsused\glslabel
8180     {%

```

Subsequent use:

```

8181     \glsifplural
8182     {%

```

Subsequent plural form:

```

8183     \glscapscase
8184     {%

```

Subsequent plural form, don't adjust case:

```

8185     \glxtrsubsequentplfmt{\glslabel}{\glsinsert}%
8186     }%
8187     {%

```

Subsequent plural form, make first letter upper case:

```

8188     \Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%
8189     }%
8190     {%

```

Subsequent plural form, all caps:

```

8191     \mfirstucMakeUppercase
8192     {\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%
8193     }%
8194     }%
8195     {%

```

Subsequent singular form

```

8196     \glscapscase
8197     {%

```

Subsequent singular form, don't adjust case:

```

8198     \glxtrsubsequentfmt{\glslabel}{\glsinsert}%
8199     }%
8200     {%

```


Subsequent singular form, make first letter upper case:

```
8201      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%  
8202      }%  
8203      {%
```

Subsequent singular form, all caps:

```
8204      \mfirstucMakeUppercase  
8205      {\glsxtrsubsequentfmt{\glslabel}{\glsinsert}}%  
8206      }%  
8207      }%  
8208      }%  
8209      {%
```

First use:

```
8210      \glsifplural  
8211      {%
```

First use plural form:

```
8212      \glscapscase  
8213      {%
```

First use plural form, don't adjust case:

```
8214      \glsxtrfullplformat{\glslabel}{\glsinsert}%  
8215      }%  
8216      {%
```

First use plural form, make first letter upper case:

```
8217      \Glsxtrfullplformat{\glslabel}{\glsinsert}%  
8218      }%  
8219      {%
```

First use plural form, all caps:

```
8220      \mfirstucMakeUppercase  
8221      {\glsxtrfullplformat{\glslabel}{\glsinsert}}%  
8222      }%  
8223      }%  
8224      {%
```

First use singular form

```
8225      \glscapscase  
8226      {%
```

First use singular form, don't adjust case:

```
8227      \glsxtrfullformat{\glslabel}{\glsinsert}%  
8228      }%  
8229      {%
```

First use singular form, make first letter upper case:

```
8230      \Glsxtrfullformat{\glslabel}{\glsinsert}%  
8231      }%  
8232      {%
```

First use singular form, all caps:

```

8233      \mfirstucMakeUppercase
8234      {\glxtrfullformat{\glslabel}{\glinsert}}}%
8235      }%
8236      }%
8237      }%
8238      }%
8239      {%

```

User supplied text.

```

8240      \glscustomtext
8241      }%
8242 }

```

trsubsequentfmt Subsequent use format (singular no case change).

```

8243 \newcommand*{\glxtrsubsequentfmt}[2]{%
8244   \glabbrvfont{\glaccessshort{#1}\ifglxtrininsertinside #2\fi}%
8245   \ifglxtrininsertinside \else#2\fi
8246 }
8247 \let\glxtrdefaultsubsequentfmt\glxtrsubsequentfmt

```

subsequentplfmt Subsequent use format (plural no case change).

```

8248 \newcommand*{\glxtrsubsequentplfmt}[2]{%
8249   \glabbrvfont{\glaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
8250   \ifglxtrininsertinside \else#2\fi
8251 }
8252 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt

```

trsubsequentfmt Subsequent use format (singular, first letter uppercase).

```

8253 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
8254   \glabbrvfont{\Glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
8255   \ifglxtrininsertinside \else#2\fi
8256 }
8257 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt

```

subsequentplfmt Subsequent use format (plural, first letter uppercase).

```

8258 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
8259   \glabbrvfont{\Glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
8260   \ifglxtrininsertinside \else#2\fi
8261 }
8262 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt

```

1.7.1 Abbreviation Styles Setup

breivationstyle

```

8263 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
8264   \ifcsundef{@glabbrv@dispstyle@setup@#2}
8265   {%

```

```

8266 \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
8267 }%
8268 {%

```

Have abbreviations already been defined for this category?

```

8269 \ifcsstring{@glsabbrv@current@#1}{#2}%
8270 {%

```

Style already set.

```

8271 }%
8272 {%
8273 \def\@glsxtr@dostylewarn{%
8274 \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
8275 {%
8276 \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
8277 style has been switched \MessageBreak
8278 for category ‘#1’, \MessageBreak
8279 but there have already been entries \MessageBreak
8280 defined for this category. Unwanted \MessageBreak
8281 side-effects may result}}%
8282 \@endfortrue
8283 }%
8284 \@glsxtr@dostylewarn

```

Set up the style for the given category.

```

8285 \csdef{@glsabbrv@current@#1}{#2}%
8286 \protected@edef\glscategorylabel{#1}%
8287 \glsxtr@applyabbrvstyle{#2}%
8288 }%
8289 }%
8290 }

```

applyabbrvstyle Apply the abbreviation style without existence check.

```

8291 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
8292 \csuse{@glsabbrv@dispstyle@setup@#1}%
8293 \csuse{@glsabbrv@dispstyle@fmts@#1}%
8294 }

```

r@applyabbrvfmt Only apply the style formats.

```

8295 \newcommand*{\glsxtr@applyabbrvfmt}[1]{%
8296 \csuse{@glsabbrv@dispstyle@fmts@#1}%
8297 }

```

abbreviationstyle This is different from `\newacronymstyle`. The first argument is the label, the second argument sets the information required when defining the new abbreviation and the third argument sets the commands used to display the full format.

```

8298 \newcommand*{\newabbreviationstyle}[3]{%
8299 \ifcsdef{@glsabbrv@dispstyle@setup@#1}
8300 {%
8301 \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already

```

```

8302     defined}{}%
8303 }%
8304 {%
8305     \csdef{@glsabbrv@dispstyle@setup@#1}{%
      Initialise hook to do nothing. The style may change this.
8306         \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
8307         #2}%
8308     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
      Assume in-line form is the same as first use. The style may change this.
8309         \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
8310         \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
8311         \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
8312         \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
      Reset \glxtrsubsequentfmt etc in case a style changes this.
8313         \let\glxtrsubsequentfmt\glxtrdefaultsubsequentfmt
8314         \let\glxtrsubsequentplfmt\glxtrdefaultsubsequentplfmt
8315         \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
8316         \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
8317         #3}%
8318     }%
8319 }

```

breivationstyle

```

8320 \newcommand*{\renewabbreviationstyle}[3]{%
8321     \ifcsundef{@glsabbrv@dispstyle@setup@#1}
8322     {%
8323         \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
8324     }%
8325     {%
8326         \csdef{@glsabbrv@dispstyle@setup@#1}{%
      Initialise hook to do nothing. The style may change this.
8327             \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
8328             #2}%
8329         \csdef{@glsabbrv@dispstyle@fmts@#1}{%
      Assume in-line form is the same as first use. The style may change this.
8330             \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
8331             \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
8332             \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
8333             \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
8334             #3}%
8335         }%
8336 }

```

breivationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

8337 \newcommand*{\letabbreviationstyle}[2]{%

```

```

8338 \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
8339 \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
8340 }

```

recated@abbrstyle

```
\@glxtr@deprecated@abbrstyle{<old-name>}{<new-name>}
```

Define a synonym for a deprecated abbreviation style.

```

8341 \newcommand*{\@glxtr@deprecated@abbrstyle}[2]{%
8342   \csdef{@glsabbrv@dispstyle@setup@#1}{%
8343     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
8344     \csuse{@glsabbrv@dispstyle@setup@#2}%
8345   }%
8346   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
8347 }

```

recatedAbbrStyle Generate warning for deprecated style use.

```

8348 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
8349   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
8350   use ‘#2’ instead}%
8351 }

```

eAbbrStyleSetup

```

8352 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
8353   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
8354   {%
8355     \PackageError{glossaries-extra}%
8356     {Unknown abbreviation style definitions ‘#1’}{}%
8357   }%
8358   {%
8359     \csname @glsabbrv@dispstyle@setup@#1\endcsname
8360   }%
8361 }

```

seAbbrStyleFmts

```

8362 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
8363   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
8364   {%
8365     \PackageError{glossaries-extra}%
8366     {Unknown abbreviation style formats ‘#1’}{}%
8367   }%
8368   {%
8369     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
8370   }%
8371 }

```

1.7.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn't set to "true". If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glstrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

`xtrinsertinside` Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```
8372 \newif\ifglstrinsertinside
8373 \glstrinsertinsidefalse
```

`trlongshortname`

```
8374 \newcommand*{\glstrlongshortname}{%
8375   \protect\glsabbrvfont{\the\glsshorttok}%
8376 }
```

`long-short`

```
8377 \newabbreviationstyle{long-short}%
8378 {%
```

Set accessibility attributes if enabled.

```
8379   \glstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
8380   \renewcommand*{\CustomAbbreviationFields}{%
8381     name={\glstrlongshortname},
8382     sort={\the\glsshorttok},
8383     first={\protect\glsfirstlongfont{\the\glslongtok}%
8384       \protect\glstrfullsep{\the\glslabeltok}%
8385       \glstrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
8386     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
8387       \protect\glstrfullsep{\the\glslabeltok}%
8388       \glstrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
8389     plural={\protect\glsabbrvfont{\the\glsshortpltok}}},%
8390     text={\protect\glsabbrvfont{\the\glsshorttok}}},%
8391     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
8392   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8393     \glshasattribute{\the\glslabeltok}{regular}%
8394     {%
8395       \glissetattribute{\the\glslabeltok}{regular}{false}%
8396     }%
8397   }
```

```

8398 }%
8399 }%
8400 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8401 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8402 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8403 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8404 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8405 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8406 \renewcommand*{\glxtrfullformat}[2]{%
8407   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8408   \ifglxtrinsertinside\else##2\fi
8409   \glxtrfullsep{##1}%
8410   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8411 }%
8412 \renewcommand*{\glxtrfullplformat}[2]{%
8413   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8414   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8415   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8416 }%
8417 \renewcommand*{\Glsxtrfullformat}[2]{%
8418   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8419   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8420   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8421 }%
8422 \renewcommand*{\Glsxtrfullplformat}[2]{%
8423   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8424   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8425   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8426 }%
8427 }

```

Set this as the default style for general abbreviations:

```

8428 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

8429 \newcommand*{\glxtrlongshortdescsort}{%
8430   \expandonce\glxtrorglong\space (\expandonce\glxtrorgshort)%
8431 }

```

ngshortdesname

```

8432 \newcommand*{\glxtrlongshortdesname}{%
8433   \protect\glslongfont{\the\glslongtok}
8434   \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
8435 }

```

long-short-desc User supplies description. The long form is included in the name.

```
8436 \newabbreviationstyle{long-short-desc}%  
8437 {%
```

Set accessibility attributes if enabled.

```
8438 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```
8439 \renewcommand*{\CustomAbbreviationFields}{%  
8440   name={\glxtrlongshortdescname},  
8441   sort={\glxtrlongshortdescsort},%  
8442   first={\protect\glsfirstlongfont{\the\glslongtok}%  
8443     \protect\glxtrfullsep{\the\glslabeltok}%  
8444     \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%  
8445   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%  
8446     \protect\glxtrfullsep{\the\glslabeltok}%  
8447     \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
```

The text key should only have the short form.

```
8448   text={\protect\glsabbrvfont{\the\glsshorttok}},%  
  
8449   plural={\protect\glsabbrvfont{\the\glsshortpltok}}%  
8450 }%
```

Unset the regular attribute if it has been set.

```
8451 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
8452   \glshasattribute{\the\glslabeltok}{regular}%  
8453   {%  
8454     \glissetattribute{\the\glslabeltok}{regular}{false}%  
8455     }%  
8456   }%  
8457 }%  
8458 }%  
8459 {%  
8460   \GlsXtrUseAbbrStyleFmts{long-short}%  
8461 }
```

trshortlongname

```
8462 \newcommand*{\glxtrshortlongname}{%  
8463   \protect\glsabbrvfont{\the\glsshorttok}%  
8464 }
```

short-long Short form followed by long form in parenthesis on first use.

```
8465 \newabbreviationstyle{short-long}%  
8466 {%
```

Set accessibility attributes if enabled.

```
8467 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
8468 \renewcommand*{\CustomAbbreviationFields}{%
```



```

8469     name={\glxtrshortlongname},
8470     sort={\the\glsshorttok},
8471     description={\the\glslongtok},%
8472     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
8473       \protect\glxtrfullsep{\the\glslabeltok}%
8474       \glxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
8475     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
8476       \protect\glxtrfullsep{\the\glslabeltok}%
8477       \glxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%

8478     text={\protect\glsabbrvfont{\the\glsshorttok}},%
8479     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8480 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8481   \glshasattribute{\the\glslabeltok}{regular}%
8482   {%
8483     \glssetattribute{\the\glslabeltok}{regular}{false}%
8484   }%
8485   {}%
8486 }%
8487 }%
8488 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8489 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8490 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8491 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8492 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8493 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8494 \renewcommand*{\glxtrfullformat}[2]{%
8495   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8496   \ifglxtrininsertinside\else##2\fi
8497   \glxtrfullsep{##1}%
8498   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8499 }%
8500 \renewcommand*{\glxtrfullplformat}[2]{%
8501   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8502   \ifglxtrininsertinside\else##2\fi
8503   \glxtrfullsep{##1}%
8504   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8505 }%
8506 \renewcommand*{\Glsxtrfullformat}[2]{%
8507   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8508   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8509   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8510 }%
8511 \renewcommand*{\Glsxtrfullplformat}[2]{%
8512   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

8513     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8514     \glxtrparen{\glsfirstlongfont{\glaccesslongpl{##1}}}%
8515 }%
8516 }

```

ortlongdescsort

```

8517 \newcommand*{\glxtrshortlongdescsort}{\the\glsshorttok}

```

ortlongdescname

```

8518 \newcommand*{\glxtrshortlongdescname}{%
8519   \protect\glsabbrvfont{\the\glsshorttok}
8520   \glxtrparen{\protect\glslongfont{\the\glslongtok}}}%
8521 }

```

short-long-desc User supplies description. The long form is included in the name.

```

8522 \newabbreviationstyle{short-long-desc}%
8523 {%

```

Set accessibility attributes if enabled.

```

8524   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

8525   \renewcommand*{\CustomAbbreviationFields}{%
8526     name={\glxtrshortlongdescname},
8527     sort={\glxtrshortlongdescsort},
8528     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
8529       \protect\glxtrfullsep{\the\glslabeltok}%
8530       \glxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
8531     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
8532       \protect\glxtrfullsep{\the\glslabeltok}%
8533       \glxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
8534     text={\protect\glsabbrvfont{\the\glsshorttok}},%
8535     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
8536 }%

```

Unset the regular attribute if it has been set.

```

8537   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8538     \glshasattribute{\the\glslabeltok}{regular}%
8539     {%
8540       \glissetattribute{\the\glslabeltok}{regular}{false}%
8541     }%
8542   }%
8543 }%
8544 }%
8545 {%
8546   \GlsXtrUseAbbrStyleFmts{short-long}%
8547 }

```

ongfootnotefont Only used by the “footnote” styles.

```
8548 \newcommand*{\glfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
8549 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%
```

trabbrvfootnote

`\glsxtrabbrvfootnote{<label>}{<long>}`

Command used by footnote abbreviation styles. The default definition ignores the first argument. The second argument `<long>` includes the font changing command and may be the singular or plural form, depending on the command that was used (for example, `\gls` or `\glspl`).

```
8550 \newcommand*{\glsxtrabbrvfootnote}[2]{\footnote{#2}}
```

xtrfootnotename

```
8551 \newcommand*{\glsxtrfootnotename}{%  
8552   \protect\glsabbrvfont{\the\glsshorttok}%  
8553 }
```

footnote Short form followed by long form in footnote on first use.

```
8554 \newabbreviationstyle{footnote}%  
8555 {%
```

Set accessibility attributes if enabled. (Add `firstshortaccess` since long form is hidden in a footnote on first use.)

```
8556   \glsxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
8557   \renewcommand*{\CustomAbbreviationFields}{%  
8558     name={\glsxtrfootnotename},  
8559     sort={\the\glsshorttok},  
8560     description={\the\glslongtok},%  
  
8561     first={\protect\glfirstabbrvfont{\the\glsshorttok}}%  
8562     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%  
8563     {\protect\glfirstlongfootnotefont{\the\glslongtok}}},%  
8564     firstplural={\protect\glfirstabbrvfont{\the\glsshortpltok}}%  
8565     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%  
8566     {\protect\glfirstlongfootnotefont{\the\glslongpltok}}},%  
  
8567     text={\protect\glsabbrvfont{\the\glsshorttok}},%  
8568     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.

```
8569   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```

8570 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8571 \glsattribute{\the\glslabeltok}{regular}%
8572 {%
8573 \glssetattribute{\the\glslabeltok}{regular}{false}%
8574 }%
8575 {}%
8576 }%
8577 }%
8578 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8579 \renewcommand*\abbrvpluralsuffix{\glsxtrabbrvpluralsuffix}%
8580 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
8581 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
8582 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8583 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

8584 \renewcommand*\glsxtrfullformat[2]{%
8585 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8586 \ifglsxtrininsertinside\else##2\fi
8587 \protect\glsxtrabbrvfootnote{##1}%
8588 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8589 }%
8590 \renewcommand*\glsxtrfullplformat[2]{%
8591 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8592 \ifglsxtrininsertinside\else##2\fi
8593 \protect\glsxtrabbrvfootnote{##1}%
8594 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8595 }%
8596 \renewcommand*\Glsxtrfullformat[2]{%
8597 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8598 \ifglsxtrininsertinside\else##2\fi
8599 \protect\glsxtrabbrvfootnote{##1}%
8600 {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8601 }%
8602 \renewcommand*\Glsxtrfullplformat[2]{%
8603 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8604 \ifglsxtrininsertinside\else##2\fi
8605 \protect\glsxtrabbrvfootnote{##1}%
8606 {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8607 }%

```

The first use full form and the inline full form use the short (long) style.

```

8608 \renewcommand*\glsxtrinlinefullformat[2]{%
8609 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8610 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8611 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8612 }%
8613 \renewcommand*\glsxtrinlinefullplformat[2]{%
8614 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%

```

```

8615 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8616 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8617 }%
8618 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8619 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8620 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8621 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
8622 }%
8623 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8624 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8625 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8626 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8627 }%
8628 }

```

short-footnote

```
8629 \letabbreviationstyle{short-footnote}{footnote}
```

ootnotedesname

```

8630 \newcommand*{\glxtrfootnotedesname}{%
8631 \protect\glssabbrvfont{\the\glssshorttok}%
8632 \protect\glxtrfullsep{\the\glslabeltok}%
8633 \protect\glxtrparen{\protect\glslongfont{\the\glslongtok}}}%
8634 }

```

ootnotedes sort

```
8635 \newcommand*{\glxtrfootnotedes sort}{\the\glssshorttok}
```

t-footnote-desc Like short-footnote but with user supplied description.

```

8636 \newabbreviationstyle{short-footnote-desc}%
8637 {%

```

Set accessibility attributes if enabled

```
8638 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```

8639 \renewcommand*{\CustomAbbreviationFields}{%
8640 name={\glxtrfootnotedesname},
8641 sort={\glxtrfootnotedes sort},
8642 first={\protect\glsfirstabbrvfont{\the\glssshorttok}%
8643 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8644 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8645 firstplural={\protect\glsfirstabbrvfont{\the\glssshortpltok}%
8646 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8647 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8648 text={\protect\glssabbrvfont{\the\glssshorttok}},%
8649 plural={\protect\glssabbrvfont{\the\glssshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

8650 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8651   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8652   \glsattribute{\the\glslabeltok}{regular}%
8653   {%
8654     \glssetattribute{\the\glslabeltok}{regular}{false}%
8655   }%
8656   {}}%
8657 }%
8658 }%
8659 {%
8660   \GlsXtrUseAbbrStyleFmts{footnote}%
8661 }

```

footnote-desc Synonym.

```
8662 \letabbreviationstyle{footnote-desc}{short-footnote-desc}
```

postfootnote Similar to footnote 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.

```

8663 \newabbreviationstyle{postfootnote}%
8664 {%

```

Set accessibility attributes if enabled. (Add `firstshortaccess` since long form is hidden in a footnote on first use.)

```
8665   \glsxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```

8666 \renewcommand*{\CustomAbbreviationFields}{%
8667   name={\glsxtrfootnotename},
8668   sort={\the\glsshorttok},
8669   description={\the\glslongtok},%
8670   first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
8671   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%

8672   text={\protect\glsabbrvfont{\the\glsshorttok}},%
8673   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.

```

8674 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8675   \csdef{glsxtrpostlink\glscategorylabel}{%
8676     \glsxtrifwasfirstuse
8677     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8678       \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}}%
8679       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
8680     }%
8681   }%

```

```

8682 }%
8683 \glshasattribute{\the\glslabeltok}{regular}%
8684 {%
8685     \glissetattribute{\the\glslabeltok}{regular}{false}%
8686 }%
8687 {}%
8688 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

8689 \renewcommand*{\glxtrsetupfulldefs}{%
8690     \let\glxtrifwasfirstuse\@secondoftwo
8691 }%
8692 }%
8693 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8694 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8695 \renewcommand*{\glabbrvfont}[1]{\glabbrvdefaultfont{##1}}%
8696 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvdefaultfont{##1}}%
8697 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongfootnotefont{##1}}%
8698 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8699 \renewcommand*{\glxtrfullformat}[2]{%
8700     \glfirstabbrvfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8701     \ifglxtrininsertinside\else##2\fi
8702 }%
8703 \renewcommand*{\glxtrfullplformat}[2]{%
8704     \glfirstabbrvfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8705     \ifglxtrininsertinside\else##2\fi
8706 }%
8707 \renewcommand*{\Glsxtrfullformat}[2]{%
8708     \glfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8709     \ifglxtrininsertinside\else##2\fi
8710 }%
8711 \renewcommand*{\Glsxtrfullplformat}[2]{%
8712     \glfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8713     \ifglxtrininsertinside\else##2\fi
8714 }%

```

The first use full form and the inline full form use the short (long) style.

```

8715 \renewcommand*{\glxtrinlinefullformat}[2]{%
8716     \glfirstabbrvfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8717     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8718     \glxtrparen{\glfirstlongfootnotefont{\glaccesslong{##1}}}%
8719 }%
8720 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8721     \glfirstabbrvfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8722     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8723     \glxtrparen{\glfirstlongfootnotefont{\glaccesslongpl{##1}}}%

```

```

8724 }%
8725 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8726   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8727   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8728   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8729 }%
8730 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8731   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8732   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8733   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8734 }%
8735 }

```

rt-postfootnote

```
8736 \letabbreviationstyle{short-postfootnote}{postfootnote}
```

stfootnote-desc Like short-postfootnote but with user supplied description.

```

8737 \newabbreviationstyle{short-postfootnote-desc}%
8738 {%

```

Set accessibility attributes if enabled.

```
8739 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```

8740 \renewcommand*{\CustomAbbreviationFields}{%
8741   name={\glxtrfootnotedesname},
8742   sort={\glxtrfootnotedesort},
8743   first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
8744   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
8745   text={\protect\glsabbrvfont{\the\glsshorttok}},%
8746   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.

```

8747 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8748   \csdef{glxtrpostlink\glscategorylabel}{%
8749     \glxtrifwasfirstuse
8750     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8751       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
8752       {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
8753     }%
8754   }%
8755 }%
8756 \glshasattribute{\the\glslabeltok}{regular}%
8757 {%
8758   \glissetattribute{\the\glslabeltok}{regular}{false}%

```



```

8759 }%
8760 {}%
8761 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

8762 \renewcommand*{\glxtrsetupfulldefs}{%
8763   \let\glxtrifwasfirstuse\@secondoftwo
8764 }%
8765 }%
8766 {%
8767   \GlsXtrUseAbbrStyleFmts{postfootnote}%
8768 }

```

stfootnote-desc

```

8769 \letabbreviationstyle{postfootnote-desc}{short-postfootnote-desc}

```

shortnolongname

```

8770 \newcommand*{\glxtrshortnolongname}{%
8771   \protect\glsabbrvfont{\the\glsshorttok}%
8772 }

```

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.

```

8773 \newabbreviationstyle{short}%
8774 {%

```

Set accessibility attributes if enabled.

```

8775   \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

8776 \renewcommand*{\CustomAbbreviationFields}{%
8777   name={\glxtrshortnolongname},
8778   sort={\the\glsshorttok},
8779   first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
8780   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
8781   text={\protect\glsabbrvfont{\the\glsshorttok}},
8782   plural={\protect\glsabbrvfont{\the\glsshortpltok}},
8783   description={\the\glslongtok}}%
8784 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8785   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8786 }%
8787 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8788 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8789 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8790 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%

```

```

8791 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8792 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8793 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8794   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
8795   \ifglsxtrininsertinside##2\fi}%
8796   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8797   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8798 }%
8799 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8800   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
8801   \ifglsxtrininsertinside##2\fi}%
8802   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8803   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8804 }%
8805 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8806   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
8807   \ifglsxtrininsertinside##2\fi}%
8808   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8809   \glsxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
8810 }%
8811 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8812   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
8813   \ifglsxtrininsertinside##2\fi}%
8814   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8815   \glsxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
8816 }%

```

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.

```

8817 \renewcommand*{\glsxtrfullformat}[2]{%
8818   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8819   \ifglsxtrininsertinside\else##2\fi
8820 }%
8821 \renewcommand*{\glsxtrfullplformat}[2]{%
8822   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8823   \ifglsxtrininsertinside\else##2\fi
8824 }%
8825 \renewcommand*{\Glsxtrfullformat}[2]{%
8826   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8827   \ifglsxtrininsertinside\else##2\fi
8828 }%
8829 \renewcommand*{\Glsxtrfullplformat}[2]{%
8830   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8831   \ifglsxtrininsertinside\else##2\fi
8832 }%
8833 }

```

Set this as the default style for acronyms:

```

8834 \setabbreviationstyle[acronym]{short}

short-nolong
8835 \letabbreviationstyle{short-nolong}{short}

rt-nolong-noreg Like short-nolong but doesn't set the regular attribute.
8836 \newabbreviationstyle{short-nolong-noreg}%
8837 {%
8838   \GlsXtrUseAbbrStyleSetup{short-nolong}%

   Unset the regular attribute if it has been set.
8839   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8840     \glshasattribute{\the\glslabeltok}{regular}%
8841     {%
8842       \glissetattribute{\the\glslabeltok}{regular}{false}%
8843     }%
8844     {}%
8845   }%
8846 }%
8847 {%
8848   \GlsXtrUseAbbrStyleFmts{short-nolong}%
8849 }

trshortdescname
8850 \newcommand*{\glxtrshortdescname}{%
8851   \protect\glsabbrvfont{\the\glsshorttok}%
8852   \protect\glxtrfullsep{\the\glslabeltok}%
8853   \protect\glxtrparen{\protect\glslongfont{\the\glslongtok}}}%
8854 }

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.
8855 \newabbreviationstyle{short-desc}%
8856 {%

   Set accessibility attributes if enabled.
8857   \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

   Setup the default fields.
8858   \renewcommand*{\CustomAbbreviationFields}{%
8859     name={\glxtrshortdescname},
8860     sort={\the\glsshorttok},
8861     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
8862     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
8863     text={\protect\glsabbrvfont{\the\glsshorttok}},
8864     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
8865   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8866     \glissetattribute{\the\glslabeltok}{regular}{true}}%
8867 }%
8868 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
8869 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8870 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8871 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8872 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8873 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short format followed by the long form in parentheses.

```
8874 \renewcommand*{\glxtrinlinefullformat}[2]{%
8875   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8876   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8877   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8878 }%
8879 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8880   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8881   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8882   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8883 }%
8884 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8885   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8886   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8887   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8888 }%
8889 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8890   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8891   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8892   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8893 }%
```

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.

```
8894 \renewcommand*{\glxtrfullformat}[2]{%
8895   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8896   \ifglxtrininsertinside\else##2\fi
8897 }%
8898 \renewcommand*{\glxtrfullplformat}[2]{%
8899   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8900   \ifglxtrininsertinside\else##2\fi
8901 }%
8902 \renewcommand*{\Glsxtrfullformat}[2]{%
8903   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8904   \ifglxtrininsertinside\else##2\fi
8905 }%
8906 \renewcommand*{\Glsxtrfullplformat}[2]{%
8907   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8908   \ifglxtrininsertinside\else##2\fi
8909 }%
8910 }
```

ort-nolong-desc

8911 \letabbreviationstyle{short-nolong-desc}{short-desc}

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

8912 \newabbreviationstyle{short-nolong-desc-noreg}%

8913 {%

8914 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%

Unset the regular attribute if it has been set.

8915 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

8916 \glshasattribute{\the\glslabeltok}{regular}%

8917 {%

8918 \glsssetattribute{\the\glslabeltok}{regular}{false}%

8919 }%

8920 {}%

8921 }%

8922 }%

8923 {%

8924 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%

8925 }

nolong-short Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

8926 \newabbreviationstyle{nolong-short}%

8927 {%

8928 \GlsXtrUseAbbrStyleSetup{short-nolong}%

8929 }%

8930 {%

8931 \GlsXtrUseAbbrStyleFmts{short-nolong}%

The inline full form displays the long form followed by the short form in parentheses.

8932 \renewcommand*{\glsxtrinlinefullformat}[2]{%

8933 \protect\glsfirstlongfont{\glssaccesslong{##1}}%

8934 \ifglsxtrininsertinside##2\fi}%

8935 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%

8936 \glsxtrparen{\glsfirstabbrvfont{\glssaccessshort{##1}}}%

8937 }%

8938 \renewcommand*{\glsxtrinlinefullplformat}[2]{%

8939 \protect\glsfirstlongfont{\glssaccesslongpl{##1}}%

8940 \ifglsxtrininsertinside##2\fi}%

8941 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%

8942 \glsxtrparen{\glsfirstabbrvfont{\glssaccessshortpl{##1}}}%

8943 }%

8944 \renewcommand*{\Glsxtrinlinefullformat}[2]{%

8945 \protect\glsfirstlongfont{\glssaccesslong{##1}}%

8946 \ifglsxtrininsertinside##2\fi}%

8947 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%

8948 \glsxtrparen{\glsfirstabbrvfont{\glssaccessshort{##1}}}%

8949 }%

8950 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%

8951 \protect\glsfirstlongfont{\glssaccesslongpl{##1}}%

```

8952      \ifglxtrinsertinside##2\fi}%
8953      \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8954      \glxtrparen{\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
8955  }%
8956 }

```

ong-short-noreg Like **nolong-short** but doesn't set the regular attribute.

```

8957 \newabbreviationstyle{nolong-short-noreg}%
8958 {%
8959   \GlsXtrUseAbbrStyleSetup{nolong-short}%

```

Unset the regular attribute if it has been set.

```

8960   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8961     \glshasattribute{\the\glslabeltok}{regular}%
8962     {%
8963       \glsselattribute{\the\glslabeltok}{regular}{false}%
8964     }%
8965     {}%
8966   }%
8967 }%
8968 {%
8969   \GlsXtrUseAbbrStyleFmts{nolong-short}%
8970 }

```

noshortdescname

```

8971 \newcommand*{\glxtrlongnoshortdescname}{%
8972   \protect\glslongfont{\the\glslongtok}%
8973 }

```

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. The accessibility attributes don't need setting here.

```

8974 \newabbreviationstyle{long-desc}%
8975 {%
8976   \renewcommand*{\CustomAbbreviationFields}{%
8977     name={\glxtrlongnoshortdescname},
8978     sort={\the\glslongtok},
8979     first={\protect\glsfirstlongfont{\the\glslongtok}},
8980     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
8981     text={\glslongfont{\the\glslongtok}},
8982     plural={\glslongfont{\the\glslongpltok}}%
8983   }%
8984   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8985     \glsselattribute{\the\glslabeltok}{regular}{true}}%
8986 }%
8987 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8988 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
8989 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
8990 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
8991 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8992 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8993 \renewcommand*\glxtrsubsequentfmt[2]{%
8994   \glslongfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8995   \ifglxtrinsertinside \else##2\fi
8996 }%
8997 \renewcommand*\glxtrsubsequentplfmt[2]{%
8998   \glslongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8999   \ifglxtrinsertinside \else##2\fi
9000 }%
9001 \renewcommand*\Glsxtrsubsequentfmt[2]{%
9002   \glslongfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9003   \ifglxtrinsertinside \else##2\fi
9004 }%
9005 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
9006   \glslongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9007   \ifglxtrinsertinside \else##2\fi
9008 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9009 \renewcommand*\glxtrinlinefullformat[2]{%
9010   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9011   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9012   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9013 }%
9014 \renewcommand*\glxtrinlinefullplformat[2]{%
9015   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9016   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9017   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
9018 }%
9019 \renewcommand*\Glsxtrinlinefullformat[2]{%
9020   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9021   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9022   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9023 }%
9024 \renewcommand*\Glsxtrinlinefullplformat[2]{%
9025   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9026   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9027   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
9028 }%

```

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.

```

9029 \renewcommand*\glxtrfullformat[2]{%
9030   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9031   \ifglxtrinsertinside\else##2\fi

```

```

9032 }%
9033 \renewcommand*{\glxtrfullplformat}[2]{%
9034   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9035   \ifglxtrinsertinside\else##2\fi
9036 }%
9037 \renewcommand*{\Glsxtrfullformat}[2]{%
9038   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9039   \ifglxtrinsertinside\else##2\fi
9040 }%
9041 \renewcommand*{\Glsxtrfullplformat}[2]{%
9042   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9043   \ifglxtrinsertinside\else##2\fi
9044 }%
9045 }

```

`ng-noshort-desc` Provide a synonym that matches similar styles.

```
9046 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

`short-desc-noreg` Like `long-noshort-desc` but doesn't set the regular attribute.

```

9047 \newabbreviationstyle{long-noshort-desc-noreg}%
9048 {%
9049   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9050   \UnsetRegularAttribute
9051   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9052     \glshasattribute{\the\glslabeltok}{regular}%
9053     \glsssetattribute{\the\glslabeltok}{regular}{false}%
9054     }%
9055     {}%
9056 }%
9057 }%
9058 {%
9059   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
9060 }

```

`longnoshortname`

```

9061 \newcommand*{\glxtrlongnoshortname}{%
9062   \protect\glsabbrvfont{\the\glsshorttok}%
9063 }

```

`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.

```

9064 \newabbreviationstyle{long}%
9065 {%
9066   \SetAccessibilityAttributesIfEnabled
9067   \GlsXtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```


Setup the default fields.

```
9067 \renewcommand*{\CustomAbbreviationFields}{%
9068   name={\glsxtrlongnoshortname},
9069   sort={\the\glsshorttok},
9070   first={\protect\glsfirstlongfont{\the\glslongtok}},
9071   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
9072   text={\glslongfont{\the\glslongtok}},
9073   plural={\glslongfont{\the\glslongpltok}},%
9074   description={\the\glslongtok}%
9075 }%
9076 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9077   \glissetattribute{\the\glslabeltok}{regular}{true}}%
9078 }%
9079 {%
9080   \GlsXtrUseAbbrStyleFmts{long-desc}%
9081 }
```

`long-noshort` Provide a synonym that matches similar styles.

```
9082 \letabbreviationstyle{long-noshort}{long}
```

`g-noshort-noreg` Like `long-noshort` but doesn't set the regular attribute.

```
9083 \newabbreviationstyle{long-noshort-noreg}%
9084 {%
9085   \GlsXtrUseAbbrStyleSetup{long-noshort}%
9086   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9087     \glshasattribute{\the\glslabeltok}{regular}%
9088     {%
9089       \glissetattribute{\the\glslabeltok}{regular}{false}%
9090     }%
9091     {}%
9092   }%
9093 }%
9094 {%
9095   \GlsXtrUseAbbrStyleFmts{long-noshort}%
9096 }
```

1.7.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

`\glsxtrscfont` Maintained for backward-compatibility.

```
9097 \newcommand*{\glsxtrscfont}[1]{\textsc{#1}}
```

`\glsabbrvscfont` Added for consistent naming.

```
9098 \newcommand*{\glsabbrvscfont}{\glsxtrscfont}
```

sxtrfirstscfont Maintained for backward-compatibility.

```
9099 \newcommand*{\glxtrfirstscfont}[1]{\glabbrvscfont{#1}}
```

irstabbrvscfont Added for consistent naming.

```
9100 \newcommand*{\glfirstabbrvscfont}{\glxtrfirstscfont}
```

and for the default short form suffix:

\glxtrscsuffix \protect needs to come inside \s to avoid interfering with all caps.

```
9101 \newcommand*{\glxtrscsuffix}{\protect\glstextup{\glxtrabbrvpluralsuffix}}
```

long-short-sc

```
9102 \newabbreviationstyle{long-short-sc}%
9103 {%
```

Set accessibility attributes if enabled.

```
9104 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
9105 \renewcommand*{\CustomAbbreviationFields}{%
9106   name={\glxtrlongshortname},
9107   sort={\the\glsshorttok},
9108   first={\protect\glfirstlongdefaultfont{\the\glslongtok}%
9109     \protect\glxtrfullsep{\the\glslabeltok}%
9110     \glxtrparen{\protect\glfirstabbrvscfont{\the\glsshorttok}}},%
9111   firstplural={\protect\glfirstlongdefaultfont{\the\glslongpltok}%
9112     \protect\glxtrfullsep{\the\glslabeltok}%
9113     \glxtrparen{\protect\glfirstabbrvscfont{\the\glsshortpltok}}},%
9114   text={\protect\glabbrvscfont{\the\glsshorttok}},%
9115   plural={\protect\glabbrvscfont{\the\glsshortpltok}},%
9116   description={\the\glslongtok}}%
9117 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9118   \glshasattribute{\the\glslabeltok}{regular}%
9119   {%
9120     \glissetattribute{\the\glslabeltok}{regular}{false}%
9121   }%
9122   {}}%
9123 }%
9124 }%
9125 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
9126 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9127 \renewcommand*{\glabbrvfont}[1]{\glabbrvscfont{##1}}%
9128 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvscfont{##1}}%
```

Use the default long fonts.

```
9129 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
9130 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline form are the same for this style.

```

9131 \renewcommand*{\glxtrfullformat}[2]{%
9132   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9133   \ifglxtrinsertinside\else##2\fi
9134   \glxtrfullsep{##1}%
9135   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9136 }%
9137 \renewcommand*{\glxtrfullplformat}[2]{%
9138   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9139   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9140   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9141 }%
9142 \renewcommand*{\Glsxtrfullformat}[2]{%
9143   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9144   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9145   \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
9146 }%
9147 \renewcommand*{\Glsxtrfullplformat}[2]{%
9148   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9149   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9150   \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
9151 }%
9152 }

```

g-short-sc-desc

```

9153 \newabbreviationstyle{long-short-sc-desc}%
9154 {%

```

Set accessibility attributes if enabled.

```

9155   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

9156 \renewcommand*{\CustomAbbreviationFields}{%
9157   name={\glxtrlongshortdescname},
9158   sort={\glxtrlongshortdescsort},%
9159   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
9160     \protect\glxtrfullsep{\the\glslabeltok}%
9161     \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
9162   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
9163     \protect\glxtrfullsep{\the\glslabeltok}%
9164     \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
9165   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9166   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
9167 }%

```

Unset the regular attribute if it has been set.

```

9168 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9169   \glshasattribute{\the\glslabeltok}{regular}%
9170   {%
9171     \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

9172 }%
9173 {}%
9174 }%
9175 }%
9176 {%

```

As long-short-sc style:

```

9177 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
9178 }

```

short-sc-long Now the short (long) version

```

9179 \newabbreviationstyle{short-sc-long}%
9180 {%

```

Set accessibility attributes if enabled.

```

9181 \glstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

9182 \renewcommand*{\CustomAbbreviationFields}{%
9183   name={\glstrshortlongname},
9184   sort={\the\glsshorttok},
9185   description={\the\gslongtok},%
9186   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
9187     \protect\glstrfullsep{\the\glslabeltok}%
9188     \glstrparen{\protect\glsfirstlongdefaultfont{\the\gslongtok}}},%
9189   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
9190     \protect\glstrfullsep{\the\glslabeltok}%
9191     \glstrparen{\protect\glsfirstlongdefaultfont{\the\gslongpltok}}},%
9192   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9193   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}

```

Unset the regular attribute if it has been set.

```

9194 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9195   \glshasattribute{\the\glslabeltok}{regular}%
9196   {%
9197     \glissetattribute{\the\glslabeltok}{regular}{false}%
9198   }%
9199   {}%
9200 }%
9201 }%
9202 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9203 \renewcommand*{\abbrvpluralsuffix}{\glstrscsuffix}%
9204 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9205 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9206 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9207 \renewcommand*{\gslongfont}[1]{\gslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9208 \renewcommand*{\glstrfullformat}[2]{%
9209   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglstrinsertinside##2\fi}%

```

```

9210 \ifglxtrinsertinside\else##2\fi
9211 \glxtrfullsep{##1}%
9212 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
9213 }%
9214 \renewcommand*{\glxtrfullplformat}[2]{%
9215 \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9216 \ifglxtrinsertinside\else##2\fi
9217 \glxtrfullsep{##1}%
9218 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
9219 }%
9220 \renewcommand*{\Glsxtrfullformat}[2]{%
9221 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9222 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9223 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
9224 }%
9225 \renewcommand*{\Glsxtrfullplformat}[2]{%
9226 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9227 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9228 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
9229 }%
9230 }

```

rt-sc-long-desc As before but user provides description

```

9231 \newabbreviationstyle{short-sc-long-desc}%
9232 {%

```

Set accessibility attributes if enabled.

```
9233 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```

9234 \renewcommand*{\CustomAbbreviationFields}{%
9235 name={\glxtrshortlongdescname},
9236 sort={\glxtrshortlongdescsort},
9237 first={\protect\glsfirstabbrvscfont{\the\glssshorttok}}%
9238 \protect\glxtrfullsep{\the\glslabeltok}}%
9239 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
9240 firstplural={\protect\glsfirstabbrvscfont{\the\glssshortpltok}}%
9241 \protect\glxtrfullsep{\the\glslabeltok}}%
9242 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
9243 text={\protect\glssabbrvscfont{\the\glssshorttok}},%
9244 plural={\protect\glssabbrvscfont{\the\glssshortpltok}}}%
9245 }%

```

Unset the regular attribute if it has been set.

```

9246 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9247 \glshasattribute{\the\glslabeltok}{regular}%
9248 {%
9249 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9250 }%
9251 {}%

```

```

9252 }%
9253 }%
9254 {%

```

As short-sc-long style:

```

9255 \GlsXtrUseAbbrStyleFmts{short-sc-long}%
9256 }

```

short-sc

```

9257 \newabbreviationstyle{short-sc}%
9258 {%

```

Set accessibility attributes if enabled.

```

9259 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9260 \renewcommand*{\CustomAbbreviationFields}{%
9261   name={\glxtrshortnolongname},
9262   sort={\the\glsshorttok},
9263   first={\protect\glfirstabbrvscfont{\the\glsshorttok}},
9264   firstplural={\protect\glfirstabbrvscfont{\the\glsshortpltok}},
9265   text={\protect\glabbrvscfont{\the\glsshorttok}},
9266   plural={\protect\glabbrvscfont{\the\glsshortpltok}},
9267   description={\the\glslongtok}}%
9268 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9269   \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9270 }%
9271 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9272 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9273 \renewcommand*{\glabbrvfont}[1]{\glabbrvscfont{##1}}%
9274 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvscfont{##1}}%
9275 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
9276 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

9277 \renewcommand*{\glxtrinlinefullformat}[2]{%
9278   \protect\glfirstabbrvscfont{\glsaccessshort{##1}}%
9279   \ifglxtrinsertinside##2\fi}%
9280   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9281   \glxtrparen{\glfirstlongdefaultfont{\glsaccesslong{##1}}}%
9282 }%
9283 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9284   \protect\glfirstabbrvscfont{\glsaccessshortpl{##1}}%
9285   \ifglxtrinsertinside##2\fi}%
9286   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9287   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9288 }%
9289 \renewcommand*{\Glsxtrinlinefullformat}[2]{%

```

```

9290 \protect\glsfirstabbrvscfont{\Glsaccesssshort{##1}%
9291 \ifglsxtrinsertinside##2\fi}%
9292 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9293 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9294 }%
9295 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9296 \protect\glsfirstabbrvscfont{\Glsaccesssshortpl{##1}%
9297 \ifglsxtrinsertinside##2\fi}%
9298 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9299 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9300 }%

```

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.

```

9301 \renewcommand*{\glsxtrfullformat}[2]{%
9302 \glsfirstabbrvscfont{\glsaccesssshort{##1}\ifglsxtrinsertinside##2\fi}%
9303 \ifglsxtrinsertinside\else##2\fi
9304 }%
9305 \renewcommand*{\glsxtrfullplformat}[2]{%
9306 \glsfirstabbrvscfont{\glsaccesssshortpl{##1}\ifglsxtrinsertinside##2\fi}%
9307 \ifglsxtrinsertinside\else##2\fi
9308 }%
9309 \renewcommand*{\Glsxtrfullformat}[2]{%
9310 \glsfirstabbrvscfont{\glsaccesssshort{##1}\ifglsxtrinsertinside##2\fi}%
9311 \ifglsxtrinsertinside\else##2\fi
9312 }%
9313 \renewcommand*{\Glsxtrfullplformat}[2]{%
9314 \glsfirstabbrvscfont{\glsaccesssshortpl{##1}\ifglsxtrinsertinside##2\fi}%
9315 \ifglsxtrinsertinside\else##2\fi
9316 }%
9317 }

```

short-sc-nolong

```

9318 \letabbreviationstyle{short-sc-nolong}{short-sc}

```

short-sc-desc

```

9319 \newabbreviationstyle{short-sc-desc}{%
9320 {%

```

Set accessibility attributes if enabled.

```

9321 \glsxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

9322 \renewcommand*{\CustomAbbreviationFields}{%
9323 name={\glsxtrshortdescname},
9324 sort={\the\glsshorttok},
9325 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
9326 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
9327 text={\protect\glsabbrvscfont{\the\glsshorttok}},
9328 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

```

9329 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9330   \glssetattribute{\the\glslabeltok}{regular}{true}}%
9331 }%
9332 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9333 \renewcommand*{\abbrvpluralsuffix}{\glstrscsuffix}%
9334 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9335 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9336 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9337 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

9338 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9339   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9340   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9341   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9342 }%
9343 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9344   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9345   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9346   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9347 }%
9348 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9349   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9350   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9351   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9352 }%
9353 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9354   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9355   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9356   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9357 }%

```

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.

```

9358 \renewcommand*{\glsxtrfullformat}[2]{%
9359   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9360   \ifglsxtrininsertinside\else##2\fi
9361 }%
9362 \renewcommand*{\glsxtrfullplformat}[2]{%
9363   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9364   \ifglsxtrininsertinside\else##2\fi
9365 }%
9366 \renewcommand*{\Glsxtrfullformat}[2]{%
9367   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9368   \ifglsxtrininsertinside\else##2\fi
9369 }%
9370 \renewcommand*{\Glsxtrfullplformat}[2]{%
9371   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9372   \ifglsxtrininsertinside\else##2\fi

```



```
9373 }%
9374 }
```

-sc-nolong-desc

```
9375 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}
```

nolong-short-sc

```
9376 \newabbreviationstyle{nolong-short-sc}%
9377 {%
9378   \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%
9379 }%
9380 {%
9381   \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%
```

The inline full form displays the long form followed by the short form in parentheses.

```
9382 \renewcommand*{\glxtrinlinefullformat}[2]{%
9383   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
9384     \ifglxtrinsertinside##2\fi}%
9385   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9386   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9387 }%
9388 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9389   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
9390     \ifglxtrinsertinside##2\fi}%
9391   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9392   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9393 }%
9394 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9395   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
9396     \ifglxtrinsertinside##2\fi}%
9397   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9398   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9399 }%
9400 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9401   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
9402     \ifglxtrinsertinside##2\fi}%
9403   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9404   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9405 }%
9406 }
```

long-noshort-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glxtrshort`. No accessibility attributes needed here.

```
9407 \newabbreviationstyle{long-noshort-sc}%
9408 {%
9409   \renewcommand*{\CustomAbbreviationFields}{%
9410     name={\glxtrlongnoshortname},
9411     sort={\the\glsshorttok},
9412     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
```

```

9413   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
9414   text={\protect\glslongdefaultfont{\the\glslongtok}},
9415   plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
9416   description={\the\glslongtok}%
9417 }%
9418 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9419   \glssetattribute{\the\glslabeltok}{regular}{true}}%
9420 }%
9421 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9422 \renewcommand*{\abbrvpluralsuffix}{\glsxtrscsuffix}%
9423 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9424 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9425 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9426 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9427 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9428   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9429   \ifglsxtrininsertinside \else##2\fi
9430 }%
9431 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9432   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9433   \ifglsxtrininsertinside \else##2\fi
9434 }%
9435 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9436   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9437   \ifglsxtrininsertinside \else##2\fi
9438 }%
9439 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9440   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9441   \ifglsxtrininsertinside \else##2\fi
9442 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9443 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9444   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9445   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9446   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9447 }%
9448 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9449   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9450   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9451   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9452 }%
9453 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9454   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9455   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9456   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9457 }%

```

```

9458 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9459   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9460   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9461   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9462 }%

```

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.

```

9463 \renewcommand*{\glxtrfullformat}[2]{%
9464   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9465   \ifglxtrinsertinside\else##2\fi
9466 }%
9467 \renewcommand*{\glxtrfullplformat}[2]{%
9468   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9469   \ifglxtrinsertinside\else##2\fi
9470 }%
9471 \renewcommand*{\Glsxtrfullformat}[2]{%
9472   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9473   \ifglxtrinsertinside\else##2\fi
9474 }%
9475 \renewcommand*{\Glsxtrfullplformat}[2]{%
9476   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9477   \ifglxtrinsertinside\else##2\fi
9478 }%
9479 }

```

long-sc Backward compatibility:

```

9480 \@glxtr@deprecated@abbrstyle{long-sc}{long-noshort-sc}

```

noshort-sc-desc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

9481 \newabbreviationstyle{long-noshort-sc-desc}%
9482 {%
9483   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9484 }%
9485 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9486 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9487 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9488 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9489 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9490 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9491 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9492   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9493   \ifglxtrinsertinside \else##2\fi
9494 }%
9495 \renewcommand*{\glxtrsubsequentplfmt}[2]{%

```

```

9496 \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9497 \ifglxtrinsertinside \else##2\fi
9498 }%
9499 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9500 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9501 \ifglxtrinsertinside \else##2\fi
9502 }%
9503 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9504 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9505 \ifglxtrinsertinside \else##2\fi
9506 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9507 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9508 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9509 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9510 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9511 }%
9512 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9513 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9514 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9515 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9516 }%
9517 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9518 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9519 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9520 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9521 }%
9522 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9523 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9524 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9525 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9526 }%

```

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.

```

9527 \renewcommand*{\glsxtrfullformat}[2]{%
9528 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9529 \ifglxtrinsertinside\else##2\fi
9530 }%
9531 \renewcommand*{\glsxtrfullplformat}[2]{%
9532 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9533 \ifglxtrinsertinside\else##2\fi
9534 }%
9535 \renewcommand*{\Glsxtrfullformat}[2]{%
9536 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9537 \ifglxtrinsertinside\else##2\fi
9538 }%
9539 \renewcommand*{\Glsxtrfullplformat}[2]{%
9540 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%

```

```

9541 \ifglxtrinsertinside\else##2\fi
9542 }%
9543 }

```

long-desc-sc Backward compatibility:

```

9544 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

9545 \newabbreviationstyle{short-sc-footnote}%
9546 {%

```

Set accessibility attributes if enabled.

```

9547 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9548 \renewcommand*{\CustomAbbreviationFields}{%
9549   name={\glxtrfootnotename},
9550   sort={\the\glsshorttok},
9551   description={\the\glslongtok},%
9552   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
9553     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9554     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9555   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
9556     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9557     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9558   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9559   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.

```

9560 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9561   \glsetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9562   \glshasattribute{\the\glslabeltok}{regular}%
9563   {%
9564     \glsetattribute{\the\glslabeltok}{regular}{false}%
9565   }%
9566   {}%
9567 }%
9568 }%
9569 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9570 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9571 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9572 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9573 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9574 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9575 \renewcommand*{\glxtrfullformat}[2]{%
9576   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%

```

```

9577 \ifglxtrinsertinside\else##2\fi
9578 \protect\glxtrabbrvfootnote{##1}%
9579 {\glsfirslongfootnotefont{\glsaccesslong{##1}}}%
9580 }%
9581 \renewcommand*{\glxtrfullplformat}[2]{%
9582 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9583 \ifglxtrinsertinside\else##2\fi
9584 \protect\glxtrabbrvfootnote{##1}%
9585 {\glsfirslongfootnotefont{\glsaccesslongpl{##1}}}%
9586 }%
9587 \renewcommand*{\Glsxtrfullformat}[2]{%
9588 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9589 \ifglxtrinsertinside\else##2\fi
9590 \protect\glxtrabbrvfootnote{##1}%
9591 {\glsfirslongfootnotefont{\glsaccesslong{##1}}}%
9592 }%
9593 \renewcommand*{\Glsxtrfullplformat}[2]{%
9594 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9595 \ifglxtrinsertinside\else##2\fi
9596 \protect\glxtrabbrvfootnote{##1}%
9597 {\glsfirslongfootnotefont{\glsaccesslongpl{##1}}}%
9598 }%

```

The first use full form and the inline full form use the short (long) style.

```

9599 \renewcommand*{\glxtrinlinfullformat}[2]{%
9600 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9601 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9602 \glxtrparen{\glsfirslongfootnotefont{\glsaccesslong{##1}}}%
9603 }%
9604 \renewcommand*{\glxtrinlinfullplformat}[2]{%
9605 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9606 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9607 \glxtrparen{\glsfirslongfootnotefont{\glsaccesslongpl{##1}}}%
9608 }%
9609 \renewcommand*{\Glsxtrinlinfullformat}[2]{%
9610 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9611 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9612 \glxtrparen{\glsfirslongfootnotefont{\glsaccesslong{##1}}}%
9613 }%
9614 \renewcommand*{\Glsxtrinlinfullplformat}[2]{%
9615 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9616 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9617 \glxtrparen{\glsfirslongfootnotefont{\glsaccesslongpl{##1}}}%
9618 }%
9619 }

```

footnote-sc Backward compatibility:

```
9620 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}
```

c-footnote-desc Like short-sc-footnote but with user supplied description.

```

9621 \newabbreviationstyle{short-sc-footnote-desc}%
9622 {%

```

Set accessibility attributes if enabled.

```

9623 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

9624 \renewcommand*{\CustomAbbreviationFields}{%
9625   name={\glxtrfootnotedescname},
9626   sort={\glxtrfootnotedescsort},
9627   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
9628     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9629     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9630   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
9631     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9632     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9633   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9634   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.

```

9635 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9636   \glsssetAttribute{\the\glslabeltok}{nohyperfirst}{true}%
9637   \glshasattribute{\the\glslabeltok}{regular}%
9638   {%
9639     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9640   }%
9641   {}%
9642 }%
9643 }%
9644 {%
9645   \GlsXtrUseAbbrStyleFmts{short-sc-footnote}%
9646 }

```

sc-postfootnote

```

9647 \newabbreviationstyle{short-sc-postfootnote}%
9648 {%

```

Set accessibility attributes if enabled.

```

9649 \glxtrAccSuppAbbrSetNameNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9650 \renewcommand*{\CustomAbbreviationFields}{%
9651   name={\glxtrfootnotename},
9652   sort={\the\glsshorttok},
9653   description={\the\glslongtok},%
9654   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
9655   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
9656   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9657   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.

```
9658 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9659 \csdef{glxstrpostlink\glscategorylabel}{%
9660 \glxtrifwasfirstuse
9661 }
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
9662 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9663 {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
9664 }%
9665 {}%
9666 }%
9667 \glshasattribute{\the\glslabeltok}{regular}%
9668 {%
9669 \glissetattribute{\the\glslabeltok}{regular}{false}%
9670 }%
9671 {}%
9672 }%
```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
9673 \renewcommand*{\glxtrsetupfulldefs}{%
9674 \let\glxtrifwasfirstuse\@secondoftwo
9675 }%
9676 }%
9677 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
9678 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9679 \renewcommand*{\glssuffixfont}[1]{\glssuffixfont{##1}}%
9680 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvfont{##1}}%
9681 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9682 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
9683 \renewcommand*{\glxtrfullformat}[2]{%
9684 \glsfirstabbrvfont{\glssuccessshort{##1}\ifglxtrininsertinside##2\fi}%
9685 \ifglxtrininsertinside\else##2\fi
9686 }%
9687 \renewcommand*{\glxtrfullplformat}[2]{%
9688 \glsfirstabbrvfont{\glssuccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9689 \ifglxtrininsertinside\else##2\fi
9690 }%
9691 \renewcommand*{\Glsxtrfullformat}[2]{%
9692 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9693 \ifglxtrininsertinside\else##2\fi
9694 }%
9695 \renewcommand*{\Glsxtrfullplformat}[2]{%
```



```

9696 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9697 \ifglxtrinsertinside\else##2\fi
9698 }%

```

The first use full form and the inline full form use the short (long) style.

```

9699 \renewcommand*{\glxtrinlinefullformat}[2]{%
9700 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9701 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9702 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
9703 }%
9704 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9705 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9706 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9707 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
9708 }%
9709 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9710 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9711 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9712 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
9713 }%
9714 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9715 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9716 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9717 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
9718 }%
9719 }

```

postfootnote-sc Backward compatibility:

```

9720 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}

```

stfootnote-desc Like short-sc-footnote but with user supplied description.

```

9721 \newabbreviationstyle{short-sc-postfootnote-desc}%
9722 {%

```

Set accessibility attributes if enabled.

```

9723 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

9724 \renewcommand*{\CustomAbbreviationFields}{%
9725 name={\glxtrfootnotedesname},
9726 sort={\glxtrfootnotedesort},
9727 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
9728 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
9729 text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9730 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.

```

9731 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9732 \csdef{glxtrpostlink\glscategorylabel}{%
9733 \glxtrifwasfirstuse

```

```

9734      {%
Needs the specific font command here as the style may have been lost by the time the foot-
note occurs.
9735      \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9736      {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
9737      }%
9738      {}%
9739      }%
9740      \glshasattribute{\the\glslabeltok}{regular}%
9741      {%
9742      \glissetattribute{\the\glslabeltok}{regular}{false}%
9743      }%
9744      {}%
9745      }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

9746 \renewcommand*{\glxtrsetupfulldefs}{%
9747 \let\glxtrifwasfirstuse\@secondoftwo
9748 }%
9749 }%
9750 {%
9751 \GlsXtrUseAbbrStyleFmts{short-sc-postfootnote}%
9752 }

```

1.7.4 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.
9753 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}

\glsabbrvsmfont Added for consistent naming.
9754 \newcommand*{\glsabbrvsmfont}{\glxtrsmfont}

sxtrfirstsmfont Maintained for backward compatibility.
9755 \newcommand*{\glxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}

irstabbrvsmfont Added for consistent naming.
9756 \newcommand*{\glsfirstabbrvsmfont}{\glxtrfirstsmfont}

and for the default short form suffix:

\glxtrsmsuffix
9757 \newcommand*{\glxtrsmsuffix}{\glxtrabbrvpluralsuffix}

```

long-short-sm

```
9758 \newabbreviationstyle{long-short-sm}%  
9759 {%
```

Set accessibility attributes if enabled.

```
9760 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
9761 \renewcommand*{\CustomAbbreviationFields}{%  
9762   name={\glxtrlongshortname},  
9763   sort={\the\glsshorttok},  
9764   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%  
9765     \protect\glxtrfullsep{\the\glslabeltok}%  
9766     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%  
9767   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%  
9768     \protect\glxtrfullsep{\the\glslabeltok}%  
9769     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%  
9770   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%  
9771   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%  
9772   description={\the\glslongtok}}%  
9773 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
9774   \glshasattribute{\the\glslabeltok}{regular}}%  
9775   {%  
9776     \glissetattribute{\the\glslabeltok}{regular}{false}%  
9777   }%  
9778   {}%  
9779 }%  
9780 }%  
9781 {%
```

```
9782 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%  
9783 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%  
9784 \renewcommand*\abbrvpluralsuffix{\glsxtrsmsuffix}%
```

Use the default long fonts.

```
9785 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%  
9786 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9787 \renewcommand*\glxtrfullformat[2]{%  
9788   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%  
9789   \ifglxtrininsertinside\else##2\fi  
9790   \glxtrfullsep{##1}%  
9791   \glxtrparen{\glsfirstabbrvsmfont{\glssaccessshort{##1}}}%  
9792 }%  
9793 \renewcommand*\glxtrfullplformat[2]{%  
9794   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%  
9795   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
9796   \glxtrparen{\glsfirstabbrvsmfont{\glssaccessshortpl{##1}}}%  
9797 }%  
9798 \renewcommand*\Glsxtrfullformat[2]{%
```

```

9799 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9800 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9801 \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
9802 }%
9803 \renewcommand*{\Glsxtrfullplformat}[2]{%
9804 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9805 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9806 \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
9807 }%
9808 }

```

g-short-sm-desc

```

9809 \newabbreviationstyle{long-short-sm-desc}%
9810 {%
    Set accessibility attributes if enabled.
9811 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
    Setup the default fields.
9812 \renewcommand*{\CustomAbbreviationFields}{%
9813   name={\glxtrlongshortdescname},
9814   sort={\glxtrlongshortdescsort},%
9815   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
9816     \protect\glxtrfullsep{\the\glslabeltok}%
9817     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
9818   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
9819     \protect\glxtrfullsep{\the\glslabeltok}%
9820     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
9821   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
9822   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
9823 }%
    Unset the regular attribute if it has been set.
9824 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9825 \glshasattribute{\the\glslabeltok}{regular}%
9826 {%
9827 \glissetattribute{\the\glslabeltok}{regular}{false}%
9828 }%
9829 {}%
9830 }%
9831 }%
9832 {%
    As long-short-sm style:
9833 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
9834 }

```

short-sm-long Now the short (long) version

```

9835 \newabbreviationstyle{short-sm-long}%
9836 {%

```

Set accessibility attributes if enabled.

```
9837 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
9838 \renewcommand*{\CustomAbbreviationFields}{%
9839   name={\glxtrshortlongname},
9840   sort={\the\glsshorttok},
9841   description={\the\gslongtok},%
9842   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
9843     \protect\glxtrfullsep{\the\glslabeltok}%
9844     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongtok}}},%
9845   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
9846     \protect\glxtrfullsep{\the\glslabeltok}%
9847     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongpltok}}},%
9848   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
9849   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
```

Unset the regular attribute if it has been set.

```
9850 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9851   \glshasattribute{\the\glslabeltok}{regular}%
9852   {%
9853     \glssetattribute{\the\glslabeltok}{regular}{false}%
9854   }%
9855   {}%
9856 }%
9857 }%
9858 {%

9859 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
9860 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
9861 \renewcommand*\abbrvpluralsuffix{\glxtrsmsuffix}%
9862 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
9863 \renewcommand*\gslongfont[1]{\gslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9864 \renewcommand*\glxtrfullformat[2]{%
9865   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9866   \ifglxtrininsertinside\else##2\fi
9867   \glxtrfullsep{##1}%
9868   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9869 }%
9870 \renewcommand*\glxtrfullplformat[2]{%
9871   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9872   \ifglxtrininsertinside\else##2\fi
9873   \glxtrfullsep{##1}%
9874   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9875 }%
9876 \renewcommand*\Glsxtrfullformat[2]{%
9877   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9878   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
```

```

9879 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
9880 }%
9881 \renewcommand*{\Glsxtrfullplformat}[2]{%
9882 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9883 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9884 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
9885 }%
9886 }

```

rt-sm-long-desc As before but user provides description

```

9887 \newabbreviationstyle{short-sm-long-desc}%
9888 {%

```

Set accessibility attributes if enabled.

```

9889 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

9890 \renewcommand*{\CustomAbbreviationFields}{%
9891 name={\glxtrshortlongdescname},
9892 sort={\glxtrshortlongdescsort},
9893 first={\protect\glsfirstabbrvsmfont{\the\glssshorttok}%
9894 \protect\glxtrfullsep{\the\glslabeltok}%
9895 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
9896 firstplural={\protect\glsfirstabbrvsmfont{\the\glssshortpltok}%
9897 \protect\glxtrfullsep{\the\glslabeltok}%
9898 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
9899 text={\protect\glsabbrvsmfont{\the\glssshorttok}},%
9900 plural={\protect\glsabbrvsmfont{\the\glssshortpltok}}}%
9901 }%

```

Unset the regular attribute if it has been set.

```

9902 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9903 \glshasattribute{\the\glslabeltok}{regular}%
9904 {%
9905 \glissetattribute{\the\glslabeltok}{regular}{false}%
9906 }%
9907 {}%
9908 }%
9909 }%
9910 {%

```

As short-sm-long style:

```

9911 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
9912 }

```

short-sm

```

9913 \newabbreviationstyle{short-sm}%
9914 {%

```

Set accessibility attributes if enabled.

```

9915 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

9916 \renewcommand*{\CustomAbbreviationFields}{%
9917   name={\glxtrshortnolongname},
9918   sort={\the\glsshorttok},
9919   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
9920   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
9921   text={\protect\glsabbrvsmfont{\the\glsshorttok}},
9922   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
9923   description={\the\glslongtok}}%
9924 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9925   \glssetattribute{\the\glslabeltok}{regular}{true}}%
9926 }%
9927 {%

9928 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
9929 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
9930 \renewcommand*{\abbrvpluralsuffix}{\glxtrsmsuffix}%
9931 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9932 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

9933 \renewcommand*{\glxtrinlinefullformat}[2]{%
9934   \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
9935   \ifglxtrininsertinside##2\fi}%
9936   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9937   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9938 }%
9939 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9940   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}%
9941   \ifglxtrininsertinside##2\fi}%
9942   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9943   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9944 }%

9945 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9946   \protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}%
9947   \ifglxtrininsertinside##2\fi}%
9948   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9949   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
9950 }%
9951 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9952   \protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}%
9953   \ifglxtrininsertinside##2\fi}%
9954   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9955   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
9956 }%

```

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.

```

9957 \renewcommand*{\glxtrfullformat}[2]{%

```

```

9958 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9959 \ifglxtrinsertinside\else##2\fi
9960 }%
9961 \renewcommand*{\glxtrfullplformat}[2]{%
9962 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9963 \ifglxtrinsertinside\else##2\fi
9964 }%
9965 \renewcommand*{\Glsxtrfullformat}[2]{%
9966 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9967 \ifglxtrinsertinside\else##2\fi
9968 }%
9969 \renewcommand*{\Glsxtrfullplformat}[2]{%
9970 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9971 \ifglxtrinsertinside\else##2\fi
9972 }%
9973 }

```

short-sm-nolong

```

9974 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

9975 \newabbreviationstyle{short-sm-desc}{%
9976 {%

```

Set accessibility attributes if enabled.

```

9977 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9978 \renewcommand*{\CustomAbbreviationFields}{%
9979 name={\glxtrshortdescname},
9980 sort={\the\glsshorttok},
9981 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
9982 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
9983 text={\protect\glsabbrvsmfont{\the\glsshorttok}},
9984 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
9985 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9986 \glssetattribute{\the\glslabeltok}{regular}{true}}%
9987 }%
9988 {%

```

```

9989 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
9990 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
9991 \renewcommand*{\abbrvpluralsuffix}{\glxtrsmsuffix}%
9992 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9993 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

9994 \renewcommand*{\glxtrinlinefullformat}[2]{%
9995 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9996 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%

```



```

9997 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9998 }%
9999 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10000 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
10001 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
10002 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
10003 }%
10004 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10005 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
10006 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
10007 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
10008 }%
10009 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10010 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
10011 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
10012 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
10013 }%

```

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.

```

10014 \renewcommand*{\glsxtrfullformat}[2]{%
10015 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
10016 \ifglsxtrinsertinside\else##2\fi
10017 }%
10018 \renewcommand*{\glsxtrfullplformat}[2]{%
10019 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
10020 \ifglsxtrinsertinside\else##2\fi
10021 }%
10022 \renewcommand*{\Glsxtrfullformat}[2]{%
10023 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
10024 \ifglsxtrinsertinside\else##2\fi
10025 }%
10026 \renewcommand*{\Glsxtrfullplformat}[2]{%
10027 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
10028 \ifglsxtrinsertinside\else##2\fi
10029 }%
10030 }

```

-sm-nolong-desc

```
10031 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}
```

nolong-short-sm

```

10032 \newabbreviationstyle{nolong-short-sm}%
10033 {%
10034 \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
10035 }%
10036 {%
10037 \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

10038 \renewcommand*{\glxtrinlinefullformat}[2]{%
10039   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
10040     \ifglxtrinsertinside##2\fi}%
10041   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10042   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
10043 }%
10044 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10045   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
10046     \ifglxtrinsertinside##2\fi}%
10047   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10048   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
10049 }%
10050 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10051   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
10052     \ifglxtrinsertinside##2\fi}%
10053   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10054   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
10055 }%
10056 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10057   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
10058     \ifglxtrinsertinside##2\fi}%
10059   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10060   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
10061 }%
10062 }

```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

10063 \newabbreviationstyle{long-noshort-sm}%
10064 {}%

```

Set accessibility attributes if enabled.

```

10065 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```

10066 \renewcommand*{\CustomAbbreviationFields}{%
10067   name={\glxtrlongnoshortname},
10068   sort={\the\glsshorttok},
10069   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
10070   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
10071   text={\protect\glslongdefaultfont{\the\glslongtok}},
10072   plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
10073   description={\the\glslongtok}%
10074 }%
10075 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10076   \glssetattribute{\the\glslabeltok}{regular}{true}}%
10077 }%
10078 {}%

10079 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%

```

```

10080 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
10081 \renewcommand*\glsabbrvpluralsuffix{\glsxtrsmsuffix}%
10082 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
10083 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10084 \renewcommand*\glsxtrsubsequentfmt}[2]{%
10085   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
10086   \ifglsxtrininsertinside \else##2\fi
10087 }%
10088 \renewcommand*\glsxtrsubsequentplfmt}[2]{%
10089   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
10090   \ifglsxtrininsertinside \else##2\fi
10091 }%
10092 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
10093   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
10094   \ifglsxtrininsertinside \else##2\fi
10095 }%
10096 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
10097   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
10098   \ifglsxtrininsertinside \else##2\fi
10099 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10100 \renewcommand*\glsxtrinlinefullformat}[2]{%
10101   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10102   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10103   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
10104 }%
10105 \renewcommand*\glsxtrinlinefullplformat}[2]{%
10106   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10107   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10108   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
10109 }%
10110 \renewcommand*\Glsxtrinlinefullformat}[2]{%
10111   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10112   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10113   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
10114 }%
10115 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
10116   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10117   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10118   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
10119 }%

```

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.

```

10120 \renewcommand*\glsxtrfullformat}[2]{%
10121   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10122   \ifglsxtrininsertinside\else##2\fi

```

```

10123 }%
10124 \renewcommand*{\glxtrfullplformat}[2]{%
10125   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10126   \ifglxtrinsertinside\else##2\fi
10127 }%
10128 \renewcommand*{\Glsxtrfullformat}[2]{%
10129   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10130   \ifglxtrinsertinside\else##2\fi
10131 }%
10132 \renewcommand*{\Glsxtrfullplformat}[2]{%
10133   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10134   \ifglxtrinsertinside\else##2\fi
10135 }%
10136 }

```

long-sm Backward compatibility:

```

10137 \@glxtr@deprecated@abbrstyle{long-sm}{long-noshort-sm}

```

noshort-sm-desc The smaller font will only be used if the short form is explicitly invoked through commands like \glsshort.

```

10138 \newabbreviationstyle{long-noshort-sm-desc}%
10139 {%
10140   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
10141 }%
10142 {%
10143   \renewcommand*\glssabbrvfont[1]{\glssabbrvsmfont{##1}}%
10144   \renewcommand*\glssfirstabbrvfont[1]{\glssfirstabbrvsmfont{##1}}%
10145   \renewcommand*\glssabbrvpluralsuffix{\glssxtrsmsuffix}%
10146   \renewcommand*\glssfirstlongfont[1]{\glssfirstlongdefaultfont{##1}}%
10147   \renewcommand*\glsslongfont[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10148 \renewcommand*{\glxtrsubsequentfnt}[2]{%
10149   \glsslongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
10150   \ifglxtrinsertinside \else##2\fi
10151 }%
10152 \renewcommand*{\glxtrsubsequentplfnt}[2]{%
10153   \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
10154   \ifglxtrinsertinside \else##2\fi
10155 }%
10156 \renewcommand*{\Glsxtrsubsequentfnt}[2]{%
10157   \glsslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
10158   \ifglxtrinsertinside \else##2\fi
10159 }%
10160 \renewcommand*{\Glsxtrsubsequentplfnt}[2]{%
10161   \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
10162   \ifglxtrinsertinside \else##2\fi
10163 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10164 \renewcommand*{\glxtrinlinefullformat}[2]{%
10165   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10166   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10167   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
10168 }%
10169 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10170   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10171   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10172   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
10173 }%
10174 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10175   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10176   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10177   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
10178 }%
10179 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10180   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10181   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10182   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
10183 }%

```

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.

```

10184 \renewcommand*{\glxtrfullformat}[2]{%
10185   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10186   \ifglxtrininsertinside\else##2\fi
10187 }%
10188 \renewcommand*{\glxtrfullplformat}[2]{%
10189   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10190   \ifglxtrininsertinside\else##2\fi
10191 }%
10192 \renewcommand*{\Glsxtrfullformat}[2]{%
10193   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10194   \ifglxtrininsertinside\else##2\fi
10195 }%
10196 \renewcommand*{\Glsxtrfullplformat}[2]{%
10197   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10198   \ifglxtrininsertinside\else##2\fi
10199 }%
10200 }

```

long-desc-sm Backward compatibility:

```

10201 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```

10202 \newabbreviationstyle{short-sm-footnote}%
10203 {}

```

Set accessibility attributes if enabled.

```
10204 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
10205 \renewcommand*{\CustomAbbreviationFields}{%
10206   name={\glxtrfootnotename},
10207   sort={\the\glsshorttok},
10208   description={\the\glslongtok},%
10209   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
10210     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10211     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
10212   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
10213     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10214     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
10215   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
10216   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.

```
10217 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10218   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
10219   \glsattribute{\the\glslabeltok}{regular}%
10220   {%
10221     \glssetattribute{\the\glslabeltok}{regular}{false}%
10222   }%
10223   {}%
10224 }%
10225}%
10226{%

10227 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
10228 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
10229 \renewcommand*{\abbrvpluralsuffix}{\glxtrsmsuffix}%
10230 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
10231 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
10232 \renewcommand*{\glxtrfullformat}[2]{%
10233   \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglxtrinsetinside##2\fi}%
10234   \ifglxtrinsetinside\else##2\fi
10235   \protect\glxtrabbrvfootnote{##1}%
10236   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10237 }%
10238 \renewcommand*{\glxtrfullplformat}[2]{%
10239   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglxtrinsetinside##2\fi}%
10240   \ifglxtrinsetinside\else##2\fi
10241   \protect\glxtrabbrvfootnote{##1}%
10242   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10243 }%
10244 \renewcommand*{\Glsxtrfullformat}[2]{%
```

```

10245 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10246 \ifglxtrinsertinside\else##2\fi
10247 \protect\glxtrabbrvfootnote{##1}%
10248 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10249 }%
10250 \renewcommand*{\Glsxtrfullplformat}[2]{%
10251 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10252 \ifglxtrinsertinside\else##2\fi
10253 \protect\glxtrabbrvfootnote{##1}%
10254 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10255 }%

```

The first use full form and the inline full form use the short (long) style.

```

10256 \renewcommand*{\glxtrinlinefullformat}[2]{%
10257 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10258 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10259 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10260 }%
10261 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10262 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10263 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10264 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10265 }%
10266 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10267 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10268 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10269 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10270 }%
10271 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10272 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10273 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10274 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10275 }%
10276 }

```

footnote-sm Backward compatibility:

```
10277 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}
```

m-footnote-desc Like short-footnote but with user supplied description.

```

10278 \newabbreviationstyle{short-sm-footnote-desc}%
10279 {%

```

Set accessibility attributes if enabled.

```
10280 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```

10281 \renewcommand*{\CustomAbbreviationFields}{%
10282 name={\glxtrfootnotedesname},
10283 sort={\glxtrfootnotedesort},
10284 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%

```

```

10285 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10286 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
10287 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
10288 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10289 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
10290 text={\protect\glsabbrvsmfont{\the\glsshorttok}}},%
10291 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.

```

10292 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10293 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
10294 \glshasattribute{\the\glslabeltok}{regular}%
10295 {%
10296 \glssetattribute{\the\glslabeltok}{regular}{false}%
10297 }%
10298 {}%
10299 }%
10300}%
10301{%
10302 \GlsXtrUseAbbrStyleFmts{short-sm-footnote}%
10303}

```

sm-postfootnote

```

10304 \newabbreviationstyle{short-sm-postfootnote}%
10305 {%

```

Set accessibility attributes if enabled.

```

10306 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

10307 \renewcommand*{\CustomAbbreviationFields}{%
10308 name={\glxtrfootnotename},
10309 sort={\the\glsshorttok},
10310 description={\the\glslongtok},%
10311 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
10312 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
10313 text={\protect\glsabbrvsmfont{\the\glsshorttok}}},%
10314 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.

```

10315 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10316 \csdef{glxtrpostlink\glscategorylabel}{%
10317 \glxtrifwasfirstuse
10318 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

10319 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
10320 {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%

```



```

10321     }%
10322     {}%
10323     }%
10324     \glshasattribute{\the\glslabeltok}{regular}%
10325     {%
10326         \glissetattribute{\the\glslabeltok}{regular}{false}%
10327     }%
10328     {}%
10329     }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

10330 \renewcommand*\glxtrsetupfulldefs{%
10331     \let\glxtrifwasfirstuse\@secondoftwo
10332 }%
10333}%
10334{%
10335 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
10336 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
10337 \renewcommand*\{abbrvpluralsuffix}{\glxtrsmsuffix}%
10338 \renewcommand*\{glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
10339 \renewcommand*\{glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

10340 \renewcommand*\glxtrfullformat[2]{%
10341     \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10342     \ifglxtrininsertinside\else##2\fi
10343 }%
10344 \renewcommand*\glxtrfullplformat[2]{%
10345     \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10346     \ifglxtrininsertinside\else##2\fi
10347 }%
10348 \renewcommand*\Glsxtrfullformat[2]{%
10349     \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10350     \ifglxtrininsertinside\else##2\fi
10351 }%
10352 \renewcommand*\Glsxtrfullplformat[2]{%
10353     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10354     \ifglxtrininsertinside\else##2\fi
10355 }%

```

The first use full form and the inline full form use the short (long) style.

```

10356 \renewcommand*\glxtrininlinefullformat[2]{%
10357     \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10358     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10359     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
10360 }%
10361 \renewcommand*\glxtrininlinefullplformat[2]{%
10362     \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10363     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

10364 \glstrparen{\glfirstlongfootnotefont{\glaccesslongpl{##1}}}%
10365 }%
10366 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10367 \glfirstabbrvsmfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
10368 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
10369 \glstrparen{\glfirstlongfootnotefont{\glaccesslong{##1}}}%
10370 }%
10371 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10372 \glfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
10373 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
10374 \glstrparen{\glfirstlongfootnotefont{\glaccesslongpl{##1}}}%
10375 }%
10376 }

```

postfootnote-sm Backward compatibility:

```

10377 \@glstr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

stfootnote-desc Like short-sm-postfootnote but with user supplied description.

```

10378 \newabbreviationstyle{short-sm-postfootnote-desc}%
10379 {%

```

Set accessibility attributes if enabled.

```

10380 \glstrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

10381 \renewcommand*{\CustomAbbreviationFields}{%
10382 name={\glstrfootnotedesname},
10383 sort={\glstrfootnotedesort},
10384 first={\protect\glfirstabbrvsmfont{\the\glsshorttok}},%
10385 firstplural={\protect\glfirstabbrvsmfont{\the\glsshortpltok}},%
10386 text={\protect\glabbrvsmfont{\the\glsshorttok}},%
10387 plural={\protect\glabbrvsmfont{\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.

```

10388 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10389 \csdef{glstrpostlink\glscategorylabel}{%
10390 \glstrifwasfirstuse
10391 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

10392 \glstrdopostpunc{\protect\glstrabbrvfootnote{\glslabel}%
10393 {\glfirstlongfootnotefont{\glentrylong{\glslabel}}}}%
10394 }%
10395 {}%
10396 }%
10397 \glshasattribute{\the\glslabeltok}{regular}%
10398 {%
10399 \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

10400    }%
10401    {}%
10402    }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

10403    \renewcommand*{\glxtrsetupfulldefs}{%
10404        \let\glxtrifwasfirstuse\@secondoftwo
10405    }%
10406 }%
10407 {%
10408    \GlsXtrUseAbbrStyleFmts{short-sm-postfootnote}%
10409 }

```

1.7.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```

10410 \newcommand*{\glsabbrvemfont}[1]{\emph{#1}}%

```

`\glfirstabbrvemfont`

```

10411 \newcommand*{\glfirstabbrvemfont}[1]{\glsabbrvemfont{#1}}%

```

The default short form suffix:

`\glxxtremsuffix`

```

10412 \newcommand*{\glxxtremsuffix}{\glxtrabbrvpluralsuffix}

```

`\glfirstlongemfont` Only used by the “long-em” styles.

```

10413 \newcommand*{\glfirstlongemfont}[1]{\glslongemfont{#1}}%

```

`\glslongemfont` Only used by the “long-em” styles.

```

10414 \newcommand*{\glslongemfont}[1]{\emph{#1}}%

```

`\long-short-em` The long form is just set in the default long font.

```

10415 \newabbreviationstyle{long-short-em}%
10416 {%

```

Set accessibility attributes if enabled.

```

10417    \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

10418    \renewcommand*{\CustomAbbreviationFields}{%
10419        name={\glxtrlongshortname},
10420        sort={\the\glsshorttok},
10421        first={\protect\glfirstlongdefaultfont{\the\glslongtok}%
10422            \protect\glxtrfullsep{\the\glslabeltok}%
10423            \glxtrparen{\protect\glfirstabbrvemfont{\the\glsshorttok}}},%
10424        firstplural={\protect\glfirstlongdefaultfont{\the\glslongpltok}%

```

```

10425 \protect\glxtrfullsep{\the\glslabeltok}%
10426 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
10427 text={\protect\glssabbbrvemfont{\the\glsshortttok}}},%
10428 plural={\protect\glssabbbrvemfont{\the\glsshortpltok}}},%
10429 description={\the\glslongtok}}%
10430 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10431 \glshasattribute{\the\glslabeltok}{regular}}%
10432 {%
10433 \glissetattribute{\the\glslabeltok}{regular}{false}}%
10434 }%
10435 {}%
10436 }%
10437 }%
10438 {%

10439 \renewcommand*{\glssabbbrvfont[1]{\glssabbbrvemfont{##1}}}%
10440 \renewcommand*{\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}}%
10441 \renewcommand*{\abbrvpluralsuffix}{\glstxtremsuffix}%

```

Use the default long fonts.

```

10442 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
10443 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10444 \renewcommand*{\glxtrfullformat}[2]{%
10445 \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10446 \ifglxtrininsertinside\else##2\fi
10447 \glxtrfullsep{##1}}%
10448 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}}%
10449 }%
10450 \renewcommand*{\glxtrfullplformat}[2]{%
10451 \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10452 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
10453 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}}%
10454 }%
10455 \renewcommand*{\Glsxtrfullformat}[2]{%
10456 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10457 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
10458 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}}%
10459 }%
10460 \renewcommand*{\Glsxtrfullplformat}[2]{%
10461 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10462 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
10463 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}}%
10464 }%
10465 }

```

g-short-em-desc

```

10466 \newabbreviationstyle{long-short-em-desc}%
10467 {%

```

Set accessibility attributes if enabled.

```
10468 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```
10469 \renewcommand*{\CustomAbbreviationFields}{%
10470   name={\glxtrlongshortdescname},
10471   sort={\glxtrlongshortdescsort},%
10472   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
10473     \protect\glxtrfullsep{\the\glslabeltok}%
10474     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10475   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
10476     \protect\glxtrfullsep{\the\glslabeltok}%
10477     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
10478   text={\protect\glssabbrvemfont{\the\glsshorttok}},%
10479   plural={\protect\glssabbrvemfont{\the\glsshortpltok}}}%
10480 }%
```

Unset the regular attribute if it has been set.

```
10481 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10482   \glshasattribute{\the\glslabeltok}{regular}%
10483   {%
10484     \glissetattribute{\the\glslabeltok}{regular}{false}%
10485   }%
10486   {}%
10487 }%
10488 }%
10489 {%
```

As long-short-em style:

```
10490 \GlsXtrUseAbbrStyleFmts{long-short-em}%
10491 }
```

ong-em-short-em

```
10492 \newabbreviationstyle{long-em-short-em}%
10493 {%
```

Set accessibility attributes if enabled.

```
10494 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields. \glslongemfont is used in the description since \glstdesc doesn't set the style.

```
10495 \renewcommand*{\CustomAbbreviationFields}{%
10496   name={\glxtrlongshortname},
10497   sort={\the\glsshorttok},
10498   first={\protect\glsfirstlongemfont{\the\glslongtok}%
10499     \protect\glxtrfullsep{\the\glslabeltok}%
10500     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10501   firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
10502     \protect\glxtrfullsep{\the\glslabeltok}%
10503     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
```

```

10504   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10505   plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
10506   description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10507   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10508     \glsattribute{\the\glslabeltok}{regular}%
10509     {%
10510       \glsattribute{\the\glslabeltok}{regular}{false}%
10511     }%
10512   }%
10513 }%
10514 }%
10515 {%

10516   \renewcommand*{\abbrvpluralsuffix}{\glstremsuffix}%
10517   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10518   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10519   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
10520   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10521   \renewcommand*{\glsxtrfullformat}[2]{%
10522     \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10523     \ifglsxtrininsertinside\else##2\fi
10524     \glsxtrfullsep{##1}%
10525     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10526   }%
10527   \renewcommand*{\glsxtrfullplformat}[2]{%
10528     \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10529     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10530     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10531   }%
10532   \renewcommand*{\Glsxtrfullformat}[2]{%
10533     \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10534     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10535     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10536   }%
10537   \renewcommand*{\Glsxtrfullplformat}[2]{%
10538     \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10539     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10540     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10541   }%
10542 }

```

m-short-em-desc

```

10543 \newabbreviationstyle{long-em-short-em-desc}%
10544 {%

```

Set accessibility attributes if enabled.

```

10545   \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```
10546 \renewcommand*{\CustomAbbreviationFields}{%
10547   name={\glxtrlongshortdescname},
10548   sort={\glxtrlongshortdescsort},%
10549   first={\protect\glsfirstlongemfont{\the\glslongtok}%
10550     \protect\glxtrfullsep{\the\glslabeltok}%
10551     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10552   firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
10553     \protect\glxtrfullsep{\the\glslabeltok}%
10554     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
10555   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10556   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
10557 }%
```

Unset the regular attribute if it has been set.

```
10558 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10559   \glshasattribute{\the\glslabeltok}{regular}%
10560   {%
10561     \glissetattribute{\the\glslabeltok}{regular}{false}%
10562   }%
10563   {}%
10564 }%
10565 }%
10566 {%
10567   \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
10568 }
```

short-em-long Now the short (long) version

```
10569 \newabbreviationstyle{short-em-long}%
10570 {%
```

Set accessibility attributes if enabled.

```
10571 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
10572 \renewcommand*{\CustomAbbreviationFields}{%
10573   name={\glxtrshortlongname},
10574   sort={\the\glsshorttok},
10575   description={\the\glslongtok},%
10576   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
10577     \protect\glxtrfullsep{\the\glslabeltok}%
10578     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
10579   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10580     \protect\glxtrfullsep{\the\glslabeltok}%
10581     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
10582   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10583   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
10584 }
```

Unset the regular attribute if it has been set.

```
10584 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10585   \glshasattribute{\the\glslabeltok}{regular}%
10586 }
```

```

10586   {%
10587     \glsetattribute{\the\glslabelltok}{regular}{false}%
10588   }%
10589   {}%
10590 }%
10591 }%
10592 {%

```

Mostly as short-long style:

```

10593 \renewcommand*{\abbrvpluralsuffix}{\glstxtremsuffix}%
10594 \renewcommand*{\glabbrvfont}[1]{\glabbrvemfont{##1}}%
10595 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvemfont{##1}}%
10596 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
10597 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10598 \renewcommand*{\glxtrfullformat}[2]{%
10599   \glfirstabbrvemfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10600   \ifglxtrininsertinside\else##2\fi
10601   \glxtrfullsep{##1}%
10602   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
10603 }%
10604 \renewcommand*{\glxtrfullplformat}[2]{%
10605   \glfirstabbrvemfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10606   \ifglxtrininsertinside\else##2\fi
10607   \glxtrfullsep{##1}%
10608   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
10609 }%
10610 \renewcommand*{\Glsxtrfullformat}[2]{%
10611   \glfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10612   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10613   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
10614 }%
10615 \renewcommand*{\Glsxtrfullplformat}[2]{%
10616   \glfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10617   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10618   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
10619 }%
10620 }

```

rt-em-long-desc As before but user provides description

```

10621 \newabbreviationstyle{short-em-long-desc}%
10622 {%

```

Set accessibility attributes if enabled.

```

10623 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

10624 \renewcommand*{\CustomAbbreviationFields}{%
10625   name={\glxtrshortlongdescname},
10626   sort={\glxtrshortlongdescsort},

```



```

10627 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
10628 \protect\glstrfullsep{\the\glslabeltok}%
10629 \glstrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
10630 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10631 \protect\glstrfullsep{\the\glslabeltok}%
10632 \glstrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
10633 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10634 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
10635 }%

```

Unset the regular attribute if it has been set.

```

10636 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10637 \glshasattribute{\the\glslabeltok}{regular}%
10638 {%
10639 \glissetattribute{\the\glslabeltok}{regular}{false}%
10640 }%
10641 {}%
10642 }%
10643 }%
10644 {%
10645 \GlsXtrUseAbbrStyleFmts{short-em-long}%
10646 }

```

hort-em-long-em

```

10647 \newabbreviationstyle{short-em-long-em}%
10648 {%

```

Set accessibility attributes if enabled.

```

10649 \glstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields. \glslongemfont is used in the description since \glsdesc doesn't set the style.

```

10650 \renewcommand*{\CustomAbbreviationFields}{%
10651 name={\glstrshortlongname},
10652 sort={\the\glsshorttok},
10653 description={\protect\glslongemfont{\the\glslongtok}},%
10654 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
10655 \protect\glstrfullsep{\the\glslabeltok}%
10656 \glstrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
10657 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10658 \protect\glstrfullsep{\the\glslabeltok}%
10659 \glstrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
10660 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10661 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%

```

Unset the regular attribute if it has been set.

```

10662 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10663 \glshasattribute{\the\glslabeltok}{regular}%
10664 {%
10665 \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

10666 }%
10667 {}%
10668 }%
10669 }%
10670 {%

10671 \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10672 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10673 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10674 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
10675 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10676 \renewcommand*{\glsxtrfullformat}[2]{%
10677   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
10678   \ifglsxtrininsertinside\else##2\fi
10679   \glsxtrfullsep{##1}%
10680   \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
10681 }%
10682 \renewcommand*{\glsxtrfullplformat}[2]{%
10683   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
10684   \ifglsxtrininsertinside\else##2\fi
10685   \glsxtrfullsep{##1}%
10686   \glsxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
10687 }%
10688 \renewcommand*{\Glsxtrfullformat}[2]{%
10689   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
10690   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10691   \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
10692 }%
10693 \renewcommand*{\Glsxtrfullplformat}[2]{%
10694   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
10695   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10696   \glsxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
10697 }%
10698 }

```

em-long-em-desc

```

10699 \newabbreviationstyle{short-em-long-em-desc}%
10700 {%

```

Set accessibility attributes if enabled.

```

10701 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

10702 \renewcommand*{\CustomAbbreviationFields}{%
10703   name={\glsxtrshortlongdescname},%
10704   sort={\glsxtrshortlongdescsort},%
10705   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
10706   \protect\glsxtrfullsep{\the\glslabeltok}}%
10707   \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%

```

```

10708   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10709   \protect\glsxtrfullsep{\the\glslabeltok}%
10710   \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
10711   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10712   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
10713 }%

```

Unset the regular attribute if it has been set.

```

10714 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10715   \glshasattribute{\the\glslabeltok}{regular}%
10716   {%
10717     \glissetattribute{\the\glslabeltok}{regular}{false}%
10718   }%
10719   {}%
10720 }%
10721 }%
10722 {%
10723   \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
10724 }

```

short-em

```

10725 \newabbreviationstyle{short-em}%
10726 {%

```

Set accessibility attributes if enabled.

```

10727 \glsxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

10728 \renewcommand*{\CustomAbbreviationFields}{%
10729   name={\glsxtrshortnolongname},
10730   sort={\the\glsshorttok},
10731   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
10732   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
10733   text={\protect\glsabbrvemfont{\the\glsshorttok}},
10734   plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
10735   description={\the\glslongtok}}%
10736 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10737   \glissetattribute{\the\glslabeltok}{regular}{true}}%
10738 }%
10739 {%

10740 \renewcommand*{\abbrvpluralsuffix}{\glsxtrsuffix}%
10741 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10742 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10743 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
10744 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

10745 \renewcommand*{\glxtrinlinefullformat}[2]{%
10746   \protect\glsfirstabbrvemfont{\glsaccessshort{##1}}%
10747   \ifglxtrininsertinside##2\fi}%

```

```

10748 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10749 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
10750 }%
10751 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10752 \protect\glsfirstabbrvemfont{\glssaccessshortpl{##1}%
10753 \ifglxtrinsertinside##2\fi}%
10754 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10755 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
10756 }%

10757 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10758 \protect\glsfirstabbrvemfont{\Glsaccessshort{##1}%
10759 \ifglxtrinsertinside##2\fi}%
10760 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10761 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
10762 }%
10763 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10764 \protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}%
10765 \ifglxtrinsertinside##2\fi}%
10766 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10767 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
10768 }%

```

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.

```

10769 \renewcommand*{\glxtrfullformat}[2]{%
10770 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10771 \ifglxtrinsertinside\else##2\fi
10772 }%
10773 \renewcommand*{\glxtrfullplformat}[2]{%
10774 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10775 \ifglxtrinsertinside\else##2\fi
10776 }%
10777 \renewcommand*{\Glsxtrfullformat}[2]{%
10778 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10779 \ifglxtrinsertinside\else##2\fi
10780 }%
10781 \renewcommand*{\Glsxtrfullplformat}[2]{%
10782 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10783 \ifglxtrinsertinside\else##2\fi
10784 }%
10785 }

```

short-em-nolong

```
10786 \letabbreviationstyle{short-em-nolong}{short-em}
```

short-em-desc

```

10787 \newabbreviationstyle{short-em-desc}%
10788 {}

```

Set accessibility attributes if enabled. The default name includes the long form but \glxtrshortdesname could be modified to omit the long form, so include the nameshortaccess attribute.

```
10789 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
10790 \renewcommand*{\CustomAbbreviationFields}{%
10791   name={\glxtrshortdesname},
10792   sort={\the\glsshorttok},
10793   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
10794   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
10795   text={\protect\glssabbrvemfont{\the\glsshorttok}},
10796   plural={\protect\glssabbrvemfont{\the\glsshortpltok}}}%
10797 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10798   \glsssetattribute{\the\glslabeltok}{regular}{true}}%
10799 }%
10800 {%

10801 \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10802 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
10803 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10804 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
10805 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short format followed by the long form in parentheses.

```
10806 \renewcommand*{\glxtrinlinefullformat}[2]{%
10807   \glsfirstabbrvemfont{\glssaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
10808   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10809   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
10810 }%

10811 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10812   \glsfirstabbrvemfont{\glssaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
10813   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10814   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
10815 }%

10816 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10817   \glsfirstabbrvemfont{\Glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
10818   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10819   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
10820 }%

10821 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10822   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
10823   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10824   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
10825 }%
```

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.

```
10826 \renewcommand*{\glxtrfullformat}[2]{%
10827   \glsfirstabbrvemfont{\glssaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
10828   \ifglxtrininsertinside\else##2\fi
```

```

10829 }%
10830 \renewcommand*{\glxstrfullplformat}[2]{%
10831   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
10832   \ifglxstrinsertinside\else##2\fi
10833 }%
10834 \renewcommand*{\Glsxtrfullformat}[2]{%
10835   \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxstrinsertinside##2\fi}%
10836   \ifglxstrinsertinside\else##2\fi
10837 }%
10838 \renewcommand*{\Glsxtrfullplformat}[2]{%
10839   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
10840   \ifglxstrinsertinside\else##2\fi
10841 }%
10842 }

```

-em-nolong-desc

```

10843 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

10844 \newabbreviationstyle{nolong-short-em}%
10845 {%
10846   \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
10847 }%
10848 {%
10849   \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

10850 \renewcommand*{\glxstrinlinefullformat}[2]{%
10851   \protect\glsfirstlongdefaultfont{\glssaccesslong{##1}%
10852     \ifglxstrinsertinside##2\fi}%
10853   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10854   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
10855 }%
10856 \renewcommand*{\glxstrinlinefullplformat}[2]{%
10857   \protect\glsfirstlongdefaultfont{\glssaccesslongpl{##1}%
10858     \ifglxstrinsertinside##2\fi}%
10859   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10860   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
10861 }%
10862 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10863   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
10864     \ifglxstrinsertinside##2\fi}%
10865   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10866   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
10867 }%
10868 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10869   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
10870     \ifglxstrinsertinside##2\fi}%
10871   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10872   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%

```

```

10873 }%
10874 }

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

10875 \newabbreviationstyle{long-noshort-em}%
10876 {%

```

Set accessibility attributes if enabled.

```

10877 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```

10878 \renewcommand*{\CustomAbbreviationFields}{%
10879   name={\glxtrlongnoshortname},
10880   sort={\the\glsshorttok},
10881   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
10882   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
10883   text={\protect\glslongdefaultfont{\the\glslongtok}},
10884   plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
10885   description={\the\glslongtok}%
10886 }%
10887 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10888   \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
10889 }%
10890 {%
10891   \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10892   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
10893   \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvemfont{##1}}%
10894   \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
10895   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10896 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10897   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
10898   \ifglxtrininsertinside \else##2\fi
10899 }%
10900 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10901   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
10902   \ifglxtrininsertinside \else##2\fi
10903 }%
10904 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10905   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
10906   \ifglxtrininsertinside \else##2\fi
10907 }%
10908 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10909   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
10910   \ifglxtrininsertinside \else##2\fi
10911 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10912 \renewcommand*{\glxtrinlinefullformat}[2]{%

```

```

10913 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10914 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10915 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10916 }%
10917 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10918 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10919 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10920 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10921 }%
10922 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10923 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10924 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10925 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10926 }%
10927 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10928 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10929 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10930 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10931 }%

```

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.

```

10932 \renewcommand*{\glxtrfullformat}[2]{%
10933 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10934 \ifglxtrinsertinside\else##2\fi
10935 }%
10936 \renewcommand*{\glxtrfullplformat}[2]{%
10937 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10938 \ifglxtrinsertinside\else##2\fi
10939 }%
10940 \renewcommand*{\Glsxtrfullformat}[2]{%
10941 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10942 \ifglxtrinsertinside\else##2\fi
10943 }%
10944 \renewcommand*{\Glsxtrfullplformat}[2]{%
10945 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10946 \ifglxtrinsertinside\else##2\fi
10947 }%
10948 }

```

`long-em` Backward compatibility:

```
10949 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}
```

`g-em-noshort-em` The short form is explicitly invoked through commands like `\glsshort`.

```

10950 \newabbreviationstyle{long-em-noshort-em}%
10951 {}

```

Set accessibility attributes if enabled.

```
10952 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel
```


Setup the default fields.

```

10953 \renewcommand*{\CustomAbbreviationFields}{%
10954   name={\glxstrlongnoshortname},
10955   sort={\the\glsshorttok},
10956   first={\protect\glxfirstlongemfont{\the\glslongtok}},
10957   firstplural={\protect\glxfirstlongemfont{\the\glslongpltok}},
10958   text={\protect\glslongemfont{\the\glslongtok}},
10959   plural={\protect\glslongemfont{\the\glslongpltok}},%
10960   description={\protect\glslongemfont{\the\glslongtok}}}%
10961 }%
10962 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10963   \glsetattribute{\the\glslabeltok}{regular}{true}}%
10964 }%
10965 {%

10966 \renewcommand*{\abbrvpluralsuffix}{\glxxtremsuffix}%
10967 \renewcommand*{\glxabbrvfont}[1]{\glxabbrvemfont{##1}}%
10968 \renewcommand*{\glxfirstabbrvfont}[1]{\glxfirstabbrvemfont{##1}}%
10969 \renewcommand*{\glxfirstlongfont}[1]{\glxfirstlongemfont{##1}}%
10970 \renewcommand*{\glxlongfont}[1]{\glxlongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10971 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10972   \glslongemfont{\glxaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
10973   \ifglxtrininsertinside \else##2\fi
10974 }%
10975 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10976   \glslongemfont{\glxaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
10977   \ifglxtrininsertinside \else##2\fi
10978 }%
10979 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10980   \glslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
10981   \ifglxtrininsertinside \else##2\fi
10982 }%
10983 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10984   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
10985   \ifglxtrininsertinside \else##2\fi
10986 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10987 \renewcommand*{\glxtrininlinefullformat}[2]{%
10988   \glxfirstlongemfont{\glxaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10989   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10990   \glxtrparen{\protect\glxfirstabbrvemfont{\glxaccessshort{##1}}}%
10991 }%
10992 \renewcommand*{\glxtrininlinefullplformat}[2]{%
10993   \glxfirstlongemfont{\glxaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10994   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10995   \glxtrparen{\protect\glxfirstabbrvemfont{\glxaccessshortpl{##1}}}%
10996 }%

```

```

10997 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10998   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10999   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
11000   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
11001 }%
11002 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11003   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11004   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
11005   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
11006 }%

```

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.

```

11007 \renewcommand*{\glxtrfullformat}[2]{%
11008   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11009   \ifglxtrinsertinside\else##2\fi
11010 }%
11011 \renewcommand*{\glxtrfullplformat}[2]{%
11012   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11013   \ifglxtrinsertinside\else##2\fi
11014 }%
11015 \renewcommand*{\Glsxtrfullformat}[2]{%
11016   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11017   \ifglxtrinsertinside\else##2\fi
11018 }%
11019 \renewcommand*{\Glsxtrfullplformat}[2]{%
11020   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11021   \ifglxtrinsertinside\else##2\fi
11022 }%
11023 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

11024 \newabbreviationstyle{long-em-noshort-em-noreg}%
11025 {}%

```

Set accessibility attributes if enabled.

```

11026 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```

11027 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

11028 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11029   \glshasattribute{\the\glslabeltok}{regular}%
11030   {%
11031     \glissetattribute{\the\glslabeltok}{regular}{false}%
11032   }%
11033   {}%
11034 }%
11035 }%
11036 {}%

```

```

11037 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
11038 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glssshort`.

```

11039 \newabbreviationstyle{long-noshort-em-desc}%
11040 {%
11041 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
11042 }%
11043 {%

11044 \renewcommand*{\abbrvpluralsuffix}{\glstremsuffix}%
11045 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
11046 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvemfont{##1}}%
11047 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
11048 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

11049 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
11050 \glsslongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
11051 \ifglsxtrininsertinside \else##2\fi
11052 }%
11053 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
11054 \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
11055 \ifglsxtrininsertinside \else##2\fi
11056 }%
11057 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
11058 \glsslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
11059 \ifglsxtrininsertinside \else##2\fi
11060 }%
11061 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
11062 \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
11063 \ifglsxtrininsertinside \else##2\fi
11064 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

11065 \renewcommand*{\glsxtrinlinefullformat}[2]{%
11066 \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
11067 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11068 \glsxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshort{##1}}}%
11069 }%
11070 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
11071 \glssfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
11072 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11073 \glsxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshortpl{##1}}}%
11074 }%
11075 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11076 \glssfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
11077 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11078 \glsxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshort{##1}}}%

```

```

11079 }%
11080 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11081   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11082   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
11083   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
11084 }%

```

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.

```

11085 \renewcommand*{\glxtrfullformat}[2]{%
11086   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11087   \ifglxtrinsertinside\else##2\fi
11088 }%
11089 \renewcommand*{\glxtrfullplformat}[2]{%
11090   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11091   \ifglxtrinsertinside\else##2\fi
11092 }%
11093 \renewcommand*{\Glsxtrfullformat}[2]{%
11094   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11095   \ifglxtrinsertinside\else##2\fi
11096 }%
11097 \renewcommand*{\Glsxtrfullplformat}[2]{%
11098   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11099   \ifglxtrinsertinside\else##2\fi
11100 }%
11101 }

```

long-desc-em Backward compatibility:

```

11102 \@glxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}

```

noshort-em-desc The short form is explicitly invoked through commands like `\glxtrshort`. The long form is emphasized. No accessibility attributes need to be set.

```

11103 \newabbreviationstyle{long-em-noshort-em-desc}%
11104 {%
11105   \renewcommand*{\CustomAbbreviationFields}{%
11106     name={\glxtrlongnoshortdescname},
11107     sort={\the\glslongtok},
11108     first={\protect\glsfirstlongemfont{\the\glslongtok}},
11109     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
11110     text={\glslongemfont{\the\glslongtok}},
11111     plural={\glslongemfont{\the\glslongpltok}}}%
11112   }%
11113   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11114     \glssetattribute{\the\glslabeltok}{regular}{true}}%
11115 }%
11116 {%
11117   \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
11118   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
11119   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%

```

```

11120 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
11121 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

11122 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
11123   \glslongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
11124   \ifglsxtrininsertinside \else##2\fi
11125 }%
11126 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
11127   \glslongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
11128   \ifglsxtrininsertinside \else##2\fi
11129 }%
11130 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
11131   \glslongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
11132   \ifglsxtrininsertinside \else##2\fi
11133 }%
11134 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
11135   \glslongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
11136   \ifglsxtrininsertinside \else##2\fi
11137 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

11138 \renewcommand*{\glsxtrinlinefullformat}[2]{%
11139   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
11140   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11141   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
11142 }%
11143 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
11144   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
11145   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11146   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
11147 }%
11148 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11149   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
11150   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11151   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
11152 }%
11153 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11154   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
11155   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11156   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
11157 }%

```

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.

```

11158 \renewcommand*{\glsxtrfullformat}[2]{%
11159   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
11160   \ifglsxtrininsertinside\else##2\fi
11161 }%
11162 \renewcommand*{\glsxtrfullplformat}[2]{%

```

```

11163 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11164 \ifglxtrinsertinside\else##2\fi
11165 }%
11166 \renewcommand*{\Glsxtrfullformat}[2]{%
11167 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11168 \ifglxtrinsertinside\else##2\fi
11169 }%
11170 \renewcommand*{\Glsxtrfullplformat}[2]{%
11171 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11172 \ifglxtrinsertinside\else##2\fi
11173 }%
11174 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

11175 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
11176 {%
11177 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%
11178 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11179 \glshasattribute{\the\glslabeltok}{regular}%
11180 }%
11181 \glssetattribute{\the\glslabeltok}{regular}{false}%
11182 }%
11183 {}%
11184 }%
11185 }%
11186 {%
11187 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
11188 }

```

ort-em-footnote

```

11189 \newabbreviationstyle{short-em-footnote}%
11190 {%
11191 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
11192 \renewcommand*{\CustomAbbreviationFields}{%
11193 name={\glxtrfootnotename},
11194 sort={\the\glsshorttok},
11195 description={\the\glslongtok},%
11196 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
11197 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
11198 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
11199 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
11200 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
11201 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
11202 text={\protect\glsabbrvemfont{\the\glsshorttok}}},%
11203 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.

```

11204 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11205 \glsssetattribute{\the\glslabelltok}{nohyperfirst}{true}%
11206 \glshasattribute{\the\glslabelltok}{regular}%
11207 {%
11208 \glsssetattribute{\the\glslabelltok}{regular}{false}%
11209 }%
11210 {}%
11211 }%
11212 }%
11213 {%

11214 \renewcommand*{\abbrvpluralsuffix}{\glstxtremsuffix}%
11215 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
11216 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
11217 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
11218 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

11219 \renewcommand*{\glsxtrfullformat}[2]{%
11220 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11221 \ifglsxtrininsertinside\else##2\fi
11222 \protect\glsxtrabbrvfootnote{##1}%
11223 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11224 }%
11225 \renewcommand*{\glsxtrfullplformat}[2]{%
11226 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11227 \ifglsxtrininsertinside\else##2\fi
11228 \protect\glsxtrabbrvfootnote{##1}%
11229 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11230 }%
11231 \renewcommand*{\Glsxtrfullformat}[2]{%
11232 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11233 \ifglsxtrininsertinside\else##2\fi
11234 \protect\glsxtrabbrvfootnote{##1}%
11235 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11236 }%
11237 \renewcommand*{\Glsxtrfullplformat}[2]{%
11238 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11239 \ifglsxtrininsertinside\else##2\fi
11240 \protect\glsxtrabbrvfootnote{##1}%
11241 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11242 }%

```

The first use full form and the inline full form use the short (long) style.

```

11243 \renewcommand*{\glsxtrinlinefullformat}[2]{%
11244 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11245 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11246 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%

```

```

11247 }%
11248 \renewcommand*{\glxtrinlinefullplformat}[2]{%
11249   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
11250   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
11251   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
11252 }%
11253 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11254   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
11255   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
11256   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
11257 }%
11258 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11259   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
11260   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
11261   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
11262 }%
11263 }

```

footnote-em Backward compatibility:

```
11264 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}
```

m-footnote-desc Like short-em-footnote but with user supplied description.

```

11265 \newabbreviationstyle{short-em-footnote-desc}%
11266 {%

```

Set accessibility attributes if enabled.

```
11267 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```

11268 \renewcommand*{\CustomAbbreviationFields}{%
11269   name={\glxtrfootnotedesname},
11270   sort={\glxtrfootnotedesort},
11271   first={\protect\glsfirstabbrvemfont{\the\glssshorttok}%
11272     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
11273     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
11274   firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}%
11275     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
11276     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
11277   text={\protect\glsabbrvemfont{\the\glssshorttok}},%
11278   plural={\protect\glsabbrvemfont{\the\glssshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

11279 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11280   \glsssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
11281   \glshasattribute{\the\glslabeltok}{regular}%
11282   {%
11283     \glsssetattribute{\the\glslabeltok}{regular}{false}%
11284   }%
11285   {}%

```



```

11286 }%
11287 }%
11288 {%
11289   \GlsXtrUseAbbrStyleFmts{short-em-footnote}%
11290 }

```

em-postfootnote

```

11291 \newabbreviationstyle{short-em-postfootnote}%
11292 {%

```

Set accessibility attributes if enabled.

```

11293   \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

11294   \renewcommand*{\CustomAbbreviationFields}{%
11295     name={\glxtrfootnotename},
11296     sort={\the\glsshorttok},
11297     description={\the\glslongtok},%
11298     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},%
11299     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},%
11300     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
11301     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.

```

11302   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11303     \csdef{glxtrpostlink\glscategorylabel}{%
11304       \glxtrifwasfirstuse
11305     }%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

11306     \glxtrdopostpunc{\protect\glxtrabbrvffootnote{\glslabel}%
11307       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
11308   }%
11309   {}%
11310 }%
11311 \glshasattribute{\the\glslabeltok}{regular}%
11312 {%
11313   \glissetattribute{\the\glslabeltok}{regular}{false}%
11314 }%
11315 {}%
11316 }%

```

The footnote needs to be suppressed in the inline form, so \glxtrfull must set the first use switch off.

```

11317   \renewcommand*{\glxtrsetupfulldefs}{%
11318     \let\glxtrifwasfirstuse\@secondoftwo
11319   }%
11320 }%
11321 {%

```

```

11322 \renewcommand*\abbrvpluralsuffix{\glxxtremsuffix}%
11323 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
11324 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
11325 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
11326 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

11327 \renewcommand*\glsxtrfullformat[2]{%
11328   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11329   \ifglsxtrininsertinside\else##2\fi
11330 }%
11331 \renewcommand*\glsxtrfullplformat[2]{%
11332   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11333   \ifglsxtrininsertinside\else##2\fi
11334 }%
11335 \renewcommand*\Glsxtrfullformat[2]{%
11336   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11337   \ifglsxtrininsertinside\else##2\fi
11338 }%
11339 \renewcommand*\Glsxtrfullplformat[2]{%
11340   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11341   \ifglsxtrininsertinside\else##2\fi
11342 }%

```

The first use full form and the inline full form use the short (long) style.

```

11343 \renewcommand*\glsxtrinlinefullformat[2]{%
11344   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11345   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11346   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11347 }%
11348 \renewcommand*\glsxtrinlinefullplformat[2]{%
11349   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11350   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11351   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11352 }%
11353 \renewcommand*\Glsxtrinlinefullformat[2]{%
11354   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11355   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11356   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11357 }%
11358 \renewcommand*\Glsxtrinlinefullplformat[2]{%
11359   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11360   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11361   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11362 }%
11363 }

```

postfootnote-em Backward compatibility:

```

11364 \@glsxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}

```

stfootnote-desc Like short-em-postfootnote but with user supplied description.

```
11365 \newabbreviationstyle{short-em-postfootnote-desc}%  
11366 {%
```

Set accessibility attributes if enabled.

```
11367 \glxstrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```
11368 \renewcommand*{\CustomAbbreviationFields}{%  
11369   name={\glxstrfootnotedesname},  
11370   sort={\glxstrfootnotedesort},  
11371   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},%  
11372   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},%  
11373   text={\protect\glsabbrvemfont{\the\glsshorttok}},%  
11374   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.

```
11375 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
11376   \csdef{glxtrpostlink\glscategorylabel}{%  
11377     \glxstrifwasfirstuse  
11378     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
11379       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%  
11380       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%  
11381     }%  
11382   }%  
11383 }%  
11384 \glshasattribute{\the\glslabeltok}{regular}%  
11385 {%  
11386   \glissetattribute{\the\glslabeltok}{regular}{false}%  
11387 }%  
11388 {}%  
11389 }%
```

The footnote needs to be suppressed in the inline form, so \glxtrfull must set the first use switch off.

```
11390 \renewcommand*{\glxtrsetupfulldefs}{%  
11391   \let\glxtrifwasfirstuse\@secondoftwo  
11392 }%  
11393 }%  
11394 {%  
11395   \GlsXtrUseAbbrStyleFmts{short-em-postfootnote}%  
11396 }
```

1.7.6 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:

`glxstruserfield` Default is the useri field.

```
11397 \newcommand*{\glxstruserfield}{useri}
```

`glxstruserparen` 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.

```
11398 \ifdef\glscurrentfieldvalue
11399 {
11400   \newcommand*{\glxstruserparen}[2]{%
11401     \glxstrfullsep{#2}%
11402     \glxstrparen
11403     {#1\ifglshasfield{\glxstruserfield}{#2}{, \glscurrentfieldvalue}{}}%
11404   }
11405 }
11406 {
11407   \newcommand*{\glxstruserparen}[2]{%
11408     \glxstrfullsep{#2}%
11409     \glxstrparen
11410     {#1\ifglshasfield{\glxstruserfield}{#2}{, \@glo@thisvalue}{}}%
11411   }
11412 }
```

Font used for short form:

`lsabbrvuserfont`

```
11413 \newcommand*{\glabbrvuserfont}[1]{\glabbrvdefaultfont{#1}}
```

Font used for short form on first use:

`stabbrvuserfont`

```
11414 \newcommand*{\glfirstabbrvuserfont}[1]{\glabbrvuserfont{#1}}
```

Font used for long form:

`glslonguserfont`

```
11415 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}
```

Font used for long form on first use:

`rstlonguserfont`

```
11416 \newcommand*{\glfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

`lsxtrusersuffix`

```
11417 \newcommand*{\glxtrusersuffix}{\glxtrabbrvpluralsuffix}
```

Description encapsulator.

`userdescription` The first argument is the description. The second argument is the label.

```
11418 \newcommand*{\gluserdescription}[2]{\glslonguserfont{#1}}
```

long-short-user

```
11419 \newabbreviationstyle{long-short-user}%
11420 {%
```

Set accessibility attributes if enabled.

```
11421 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
11422 \renewcommand*{\CustomAbbreviationFields}{%
11423   name={\glxtrlongshortname},
11424   sort={\the\glsshorttok},
11425   first={\protect\glsfirstlonguserfont{\the\glslongtok}%
11426     \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
11427     {\the\glslabeltok}},%
11428   firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
11429     \protect\glxtruserparen
11430     {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%

11431   text={\protect\glssabbvruserfont{\the\glsshorttok}},%
11432   plural={\protect\glssabbvruserfont{\the\glsshortpltok}},%
11433   description={\protect\glssuserdescription{\the\glslongtok}%
11434     {\the\glslabeltok}}}%
```

Unset the regular attribute if it has been set.

```
11435 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11436   \glshasattribute{\the\glslabeltok}{regular}%
11437   {%
11438     \glissetattribute{\the\glslabeltok}{regular}{false}%
11439   }%
11440   {}%
11441 }%
11442 }%
11443 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
11444 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
11445 \renewcommand*{\glssabbvrfont}[1]{\glssabbvruserfont{##1}}%
11446 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
11447 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
11448 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
11449 \renewcommand*{\glxtrfullformat}[2]{%
11450   \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11451   \ifglxtrininsertinside\else##2\fi
11452   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
11453 }%
11454 \renewcommand*{\glxtrfullplformat}[2]{%
11455   \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11456   \ifglxtrininsertinside\else##2\fi
11457   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
```

```

11458 }%
11459 \renewcommand*{\Glsxtrfullformat}[2]{%
11460   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11461   \ifglxtrinsertinside\else##2\fi
11462   \glxtruserparen{\glsfirstabbrvuserfont{\Glsaccessshort{##1}}{##1}%
11463 }%
11464 \renewcommand*{\Glsxtrfullplformat}[2]{%
11465   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11466   \ifglxtrinsertinside\else##2\fi
11467   \glxtruserparen{\glsfirstabbrvuserfont{\Glsaccessshortpl{##1}}{##1}%
11468 }%
11469 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

11470 \newabbreviationstyle{long-postshort-user}%
11471 {%

```

Set accessibility attributes if enabled.

```

11472 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

11473 \renewcommand*{\CustomAbbreviationFields}{%
11474   name={\glxtrlongshortname},
11475   sort={\the\glsshorttok},
11476   first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
11477   firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

11478   text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11479   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
11480   description={\protect\glsuserdescription{\the\glslongtok}%
11481     {\the\glslabeltok}}}%
11482 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11483   \csdef{glxtrpostlink\glscategorylabel}{%
11484     \glxtrifwasfirstuse
11485     {%
11486       \glxtruserparen
11487         {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
11488         {\glslabel}%
11489     }%
11490   }%
11491 }%
11492 \glshasattribute{\the\glslabeltok}{regular}%
11493 {%
11494   \glissetattribute{\the\glslabeltok}{regular}{false}%
11495 }%
11496 }%
11497 }%
11498 }%
11499 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11500 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
11501 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
11502 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
11503 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
11504 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

11505 \renewcommand*{\glxtrfullformat}[2]{%
11506   \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11507   \ifglxtrininsertinside\else##2\fi
11508 }%
11509 \renewcommand*{\glxtrfullplformat}[2]{%
11510   \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11511   \ifglxtrininsertinside\else##2\fi
11512 }%
11513 \renewcommand*{\Glsxtrfullformat}[2]{%
11514   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11515   \ifglxtrininsertinside\else##2\fi
11516 }%
11517 \renewcommand*{\Glsxtrfullplformat}[2]{%
11518   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11519   \ifglxtrininsertinside\else##2\fi
11520 }%

```

In-line format:

```

11521 \renewcommand*{\glxtrinlinefullformat}[2]{%
11522   \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11523   \ifglxtrininsertinside\else##2\fi
11524   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
11525 }%
11526 \renewcommand*{\glxtrinlinefullplformat}[2]{%
11527   \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11528   \ifglxtrininsertinside\else##2\fi
11529   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
11530 }%
11531 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11532   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11533   \ifglxtrininsertinside\else##2\fi
11534   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
11535 }%
11536 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11537   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11538   \ifglxtrininsertinside\else##2\fi
11539   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
11540 }%
11541 }

```

Small-caps is awkward, so support for that is added.

abbrvscuserfont

```

11542 \newcommand*{\glsabbrvscuserfont}{\glsabbrvscfont}%

```

abbrvscuserfont

```
11543 \newcommand*{\glxfirstabbrvscuserfont}{\glxabbrvscuserfont}%
```

The default short form suffix:

xtrscusersuffix

```
11544 \newcommand*{\glxtrscusersuffix}{\glxtrscsuffix}
```

lsxtrscusername The default name format for this style.

```
11545 \newcommand*{\glxtrlongshortscusername}{%
11546   \protect\glxabbrvscuserfont{\the\glsshorttok}}%
11547 }
```

stshort-sc-user Like long-short-sc-user but uses smallcaps.

```
11548 \newabbreviationstyle{long-postshort-sc-user}%
11549 {%
```

Set accessibility attributes if enabled.

```
11550   \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
11551   \renewcommand*{\CustomAbbreviationFields}{%
11552     name={\glxtrlongshortscusername},
11553     sort={\the\glsshorttok},
11554     first={\protect\glxfirstlonguserfont{\the\glslongtok}},%
11555     firstplural={\protect\glxfirstlonguserfont{\the\glslongpltok}},%
11556     text={\protect\glxabbrvscuserfont{\the\glsshorttok}},%
11557     plural={\protect\glxabbrvscuserfont{\the\glsshortpltok}},%
11558     description={\protect\glxuserdescription{\the\glslongtok}%
11559       {\the\glslabeltok}}}%
11560   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11561     \csdef{glxtrpostlink\glscategorylabel}{%
11562       \glxtrifwasfirstuse
11563       {%
11564         \glxtruserparen
11565         {\glxfirstabbrvscuserfont{\glxentryshort{\glslabel}}}%
11566         {\glslabel}%
11567       }%
11568     }%
11569   }%
11570   \glshasattribute{\the\glslabeltok}{regular}%
11571   {%
11572     \glissetattribute{\the\glslabeltok}{regular}{false}%
11573   }%
11574   {}%
11575 }%
11576 }%
11577 {%
```


In case the user wants to mix and match font styles, these are redefined here.

```

11578 \renewcommand*{\abbrvpluralsuffix}{\glxtrscusersuffix}%
11579 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscuserfont{##1}}%
11580 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscuserfont{##1}}%
11581 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
11582 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

11583 \renewcommand*{\glxtrfullformat}[2]{%
11584   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11585   \ifglxtrininsertinside\else##2\fi
11586 }%
11587 \renewcommand*{\glxtrfullplformat}[2]{%
11588   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11589   \ifglxtrininsertinside\else##2\fi
11590 }%
11591 \renewcommand*{\Glsxtrfullformat}[2]{%
11592   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11593   \ifglxtrininsertinside\else##2\fi
11594 }%
11595 \renewcommand*{\Glsxtrfullplformat}[2]{%
11596   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11597   \ifglxtrininsertinside\else##2\fi
11598 }%

```

In-line format:

```

11599 \renewcommand*{\glxtrinlinefullformat}[2]{%
11600   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11601   \ifglxtrininsertinside\else##2\fi
11602   \glxtruserparen{\glsfirstabbrvscuserfont{\glsaccessshort{##1}}}{##1}%
11603 }%
11604 \renewcommand*{\glxtrinlinefullplformat}[2]{%
11605   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11606   \ifglxtrininsertinside\else##2\fi
11607   \glxtruserparen{\glsfirstabbrvscuserfont{\glsaccessshortpl{##1}}}{##1}%
11608 }%
11609 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11610   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11611   \ifglxtrininsertinside\else##2\fi
11612   \glxtruserparen{\glsfirstabbrvscuserfont{\glsaccessshort{##1}}}{##1}%
11613 }%
11614 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11615   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11616   \ifglxtrininsertinside\else##2\fi
11617   \glxtruserparen{\glsfirstabbrvscuserfont{\glsaccessshortpl{##1}}}{##1}%
11618 }%
11619 }

```

ortuserdesname

```

11620 \newcommand*{\glxtrlongshortuserdesname}{%

```

```

11621 \protect\glslonguserfont{\the\glslongtok}%
11622 \protect\glsxtruserparen
11623 {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}%
11624 }

```

`short-user-desc` Like `long-postshort-user` but the user supplies the description.

```

11625 \newabbreviationstyle{long-postshort-user-desc}%
11626 {%
    Set accessibility attributes if enabled.
11627 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
    Setup the default fields.
11628 \renewcommand*{\CustomAbbreviationFields}{%
11629     name={\glsxtrlongshortuserdescname},
11630     sort={\the\glslongtok},
11631     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
11632     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
11633     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11634     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
11635 }%
11636 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11637 \csdef{glsxtrpostlink\glscategorylabel}{%
11638     \glsxtrifwasfirstuse
11639     {%
11640         \glsxtruserparen
11641         {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
11642         {\glslabel}%
11643     }%
11644     }%
11645 }%
11646 \glsattribute{\the\glslabeltok}{regular}%
11647 {%
11648     \glssetattribute{\the\glslabeltok}{regular}{false}%
11649 }%
11650 {}%
11651 }%
11652 }%
11653 {%
11654 \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
11655 }

```

`tscuserdescname`

```

11656 \newcommand*{\glsxtrlongshorttscuserdescname}{%
11657 \protect\glslonguserfont{\the\glslongtok}%
11658 \protect\glsxtruserparen
11659 {\protect\glsabbrvscuserfont{\the\glsshorttok}}{\the\glslabeltok}%
11660 }

```

rt-sc-user-desc Like long-postshort-sc-user but the user supplies the description.

```
11661 \newabbreviationstyle{long-postshort-sc-user-desc}%  
11662 {%
```

Set accessibility attributes if enabled.

```
11663 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```
11664 \renewcommand*{\CustomAbbreviationFields}{%  
11665   name={\glxtrlongshortscuserdescname},  
11666   sort={\the\glslongtok},  
11667   first={\protect\glsfirstlonguserfont{\the\glslongtok}},%  
11668   firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%  
11669   text={\protect\glssabrvscuserfont{\the\glsshorttok}},%  
11670   plural={\protect\glssabrvscuserfont{\the\glsshortpltok}}}%  
11671 }%  
11672 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
11673   \csdef{glxtrpostlink\glscategorylabel}{%  
11674     \glxtrifwasfirstuse  
11675     {%  
11676       \glxtruserparen  
11677       {\glsfirstabrvscuserfont{\glsenryshort{\glslabel}}}%  
11678       {\glslabel}}%  
11679     }%  
11680     {}%  
11681   }%  
11682   \glshasattribute{\the\glslabeltok}{regular}%  
11683   {%  
11684     \glissetattribute{\the\glslabeltok}{regular}{false}%  
11685   }%  
11686   {}%  
11687 }%  
11688 }%  
11689 {%  
11690   \GlsXtrUseAbbrStyleFmts{long-postshort-sc-user}%  
11691 }
```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```
11692 \newabbreviationstyle{short-postlong-user}%  
11693 {%
```

Set accessibility attributes if enabled.

```
11694 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
11695 \renewcommand*{\CustomAbbreviationFields}{%  
11696   name={\glxtrshortlongname},  
11697   sort={\the\glsshorttok},  
11698   first={\protect\glsfirstlonguserfont{\the\glslongtok}},%  
11699   firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
```

```

11700     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11701     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
11702     description={\protect\glsuserdescription{\the\glslongtok}%
11703       {\the\glslabeltok}}}%
11704 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11705   \csdef{glsxtrpostlink\glscategorylabel}{%
11706     \glsxtrifwasfirstuse
11707     {%
11708       \glsxtruserparen
11709       {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
11710       {\glslabel}}%
11711     }%
11712   }%
11713 }%
11714 \glsattribute{\the\glslabeltok}{regular}%
11715 {%
11716   \glssetAttribute{\the\glslabeltok}{regular}{false}%
11717 }%
11718 {}%
11719 }%
11720 }%
11721 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11722 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
11723 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
11724 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
11725 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
11726 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

11727 \renewcommand*{\glsxtrfullformat}[2]{%
11728   \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11729   \ifglsxtrininsertinside\else##2\fi
11730 }%
11731 \renewcommand*{\glsxtrfullplformat}[2]{%
11732   \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11733   \ifglsxtrininsertinside\else##2\fi
11734 }%
11735 \renewcommand*{\Glsxtrfullformat}[2]{%
11736   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11737   \ifglsxtrininsertinside\else##2\fi
11738 }%
11739 \renewcommand*{\Glsxtrfullplformat}[2]{%
11740   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11741   \ifglsxtrininsertinside\else##2\fi
11742 }%

```

In-line format:

```

11743 \renewcommand*{\glsxtrinlinefullformat}[2]{%

```

```

11744 \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11745 \ifglxtrinsertinside\else##2\fi
11746 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
11747 }%
11748 \renewcommand*{\glxtrinlinefullplformat}[2]{%
11749 \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11750 \ifglxtrinsertinside\else##2\fi
11751 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11752 }%
11753 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11754 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11755 \ifglxtrinsertinside\else##2\fi
11756 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
11757 }%
11758 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11759 \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11760 \ifglxtrinsertinside\else##2\fi
11761 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11762 }%
11763 }

```

onguserdescname

```

11764 \newcommand*{\glxtrshortlonguserdescname}{%
11765 \protect\glsabbrvuserfont{\the\glsshorttok}%
11766 \protect\glxtruserparen
11767 {\protect\glslonguserfont{\the\glslongpltok}}}%
11768 {\the\glslabeltok}%
11769 }

```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```

11770 \newabbreviationstyle{short-postlong-user-desc}%
11771 {%

```

Set accessibility attributes if enabled.

```

11772 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

11773 \renewcommand*{\CustomAbbreviationFields}{%
11774 name={\glxtrshortlonguserdescname},
11775 sort={\the\glsshorttok},
11776 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
11777 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

11778 text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11779 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
11780 }%
11781 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11782 \csdef{glxtrpostlink\glscategorylabel}{%
11783 \glxtrifwasfirstuse
11784 {%

```

```

11785     \glxtruserparen
11786     {\glsfirstlonguserfont{\gl Sentrylong{\glslabel}}}%
11787     {\glslabel}%
11788   }%
11789   {}%
11790 }%
11791 \glshasattribute{\the\glslabeltok}{regular}%
11792 {%
11793   \glissetattribute{\the\glslabeltok}{regular}{false}%
11794   }%
11795   {}%
11796 }%
11797 }%
11798 {%
11799   \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
11800 }

```

short-user-desc

```

11801 \newabbreviationstyle{long-short-user-desc}%
11802 {%
11803   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
11804   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
11805   Setup the default fields.
11806   \renewcommand*{\CustomAbbreviationFields}{%
11807     name={\glxtrlongshortuserdescname},
11808     sort={\glxtrlongshortdescsort},%
11809     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
11810       \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
11811     {\the\glslabeltok}},%
11812     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
11813       \protect\glxtruserparen
11814       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
11815     text={\protect\glsabbrvfont{\the\glsshorttok}},%
11816     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
11817   }%
11818   Unset the regular attribute if it has been set.
11819   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11820     \glshasattribute{\the\glslabeltok}{regular}%
11821     {%
11822       \glissetattribute{\the\glslabeltok}{regular}{false}%
11823       }%
11824     {}%
11825   }%
11826   \GlsXtrUseAbbrStyleFmts{long-short-user}%
11827 }

```

short-long-user

```
11827 \newabbreviationstyle{short-long-user}%  
11828 {%
```

Set accessibility attributes if enabled.

```
11829 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

\glslonguserfont is used in the description since \glsdesc doesn't set the style. (Now in \glsuserdescription.)

```
11830 \renewcommand*{\CustomAbbreviationFields}{%  
11831   name={\glxtrshortlongname},  
11832   sort={\the\glsshorttok},  
11833   description={\protect\glsuserdescription{\the\glslongtok}%  
11834     {\the\glslabeltok}},%  
11835   first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%  
11836     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%  
11837     {\the\glslabeltok}},%  
11838   firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%  
11839     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%  
11840     {\the\glslabeltok}},%  
  
11841   text={\protect\glsabbrvuserfont{\the\glsshorttok}},%  
11842   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
```

Unset the regular attribute if it has been set.

```
11843 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
11844   \glshasattribute{\the\glslabeltok}{regular}%  
11845   {%  
11846     \glissetattribute{\the\glslabeltok}{regular}{false}%  
11847   }%  
11848   {}%  
11849 }%  
11850 }%  
11851 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
11852 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%  
11853 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%  
11854 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%  
11855 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%  
11856 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
11857 \renewcommand*{\glxtrfullformat}[2]{%  
11858   \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%  
11859   \ifglxtrininsertinside\else##2\fi  
11860   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%  
11861 }%  
11862 \renewcommand*{\glxtrfullplformat}[2]{%
```

```

11863 \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11864 \ifglxtrinsertinside\else##2\fi
11865 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11866 }%
11867 \renewcommand*{\Glsxtrfullformat}[2]{%
11868 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11869 \ifglxtrinsertinside\else##2\fi
11870 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
11871 }%
11872 \renewcommand*{\Glsxtrfullplformat}[2]{%
11873 \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11874 \ifglxtrinsertinside\else##2\fi
11875 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11876 }%
11877 }

```

-long-user-desc

```

11878 \newabbreviationstyle{short-long-user-desc}%
11879 {%
    Set accessibility attributes if enabled.
11880 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
    Setup the default fields.
11881 \renewcommand*{\CustomAbbreviationFields}{%
11882   name={\glxtrshortlonguserdescname},
11883   sort={\glxtrshortlongdescsort},%
11884   first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
11885   \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
11886   {\the\glslabeltok}},%
11887   firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
11888   \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%
11889   {\the\glslabeltok}},%
11890   text={\protect\glsabbrvfont{\the\glsshorttok}},%
11891   plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
11892 }%
    Unset the regular attribute if it has been set.
11893 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11894 \glshasattribute{\the\glslabeltok}{regular}%
11895 {%
11896 \glissetattribute{\the\glslabeltok}{regular}{false}%
11897 }%
11898 {}%
11899 }%
11900}%
11901{%
11902 \GlsXtrUseAbbrStyleFmts{short-long-user}%
11903 }

```


1.7.7 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 `\glstrlong` 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.

```
11904 \newrobustcmd*{\glstrifhyphenstart}[3]{%
11905   \ifx\glsinsert#1\relax
11906   \expandafter\@glstrifhyphenstart#1\relax\relax
11907   \@end@glstrifhyphenstart{#2}{#3}%
11908   \else
11909   \@glstrifhyphenstart#1\relax\relax\@end@glstrifhyphenstart{#2}{#3}%
11910   \fi
11911 }
```

`trifhyphenstart`

```
11912 \def\@glstrifhyphenstart#1#2\@end@glstrifhyphenstart#3#4{%
11913   \ifx-#1\relax#3\else #4\fi
11914 }
```

`longhyphenshort`

`\glstrlonghyphenshort{<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.

```
11915 \newcommand*{\glstrlonghyphenshort}[4]{%
```

Grouping is needed to localise the redefinitions.

```
11916 {%
```

If `<insert>` starts with a hyphen, redefine `\glstrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.) No change is made to `\glstrwordsep` if `<insert>` doesn't start with a hyphen.

```
11917   \glstrifhyphenstart{#4}{\def\glstrwordsep{-}}{}%
11918   \glsfirstlonghyphenfont{#2\ifglstrinsertinside{#4}\fi}%
11919   \ifglstrinsertinside\else{#4}\fi
11920   \glstrfullsep{#1}%
11921   \glstrparen{\glsfirstabbrvhyphenfont{#3\ifglstrinsertinside{#4}\fi}%
11922   \ifglstrinsertinside\else{#4}\fi}%
11923 }%
11924 }
```

`abbrvhyphenfont`

```
11925 \newcommand*{\glsabbrvhyphenfont}{\glsabbrvdefaultfont}%
```

abbrvhyphenfont

```
11926 \newcommand*{\glfirstabbrvhyphenfont}{\glabbrvhyphenfont}%
```

slonghyphenfont

```
11927 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%
```

tlonghyphenfont

```
11928 \newcommand*{\glfirstlonghyphenfont}{\glslonghyphenfont}%
```

The default short form suffix:

xtrhyphensuffix

```
11929 \newcommand*{\glxtrhyphensuffix}{\glxtrabbrvpluralsuffix}
```

en-short-hyphen Designed for use with the markwords attribute.

```
11930 \newabbreviationstyle{long-hyphen-short-hyphen}%
```

```
11931 {%
```

Set accessibility attributes if enabled.

```
11932 \glxtrAccSuppAbbrSetFirstLongAttr{\glscategorylabel}
```

Setup the default fields.

```
11933 \renewcommand*{\CustomAbbreviationFields}{%
11934   name={\glxtrlongshortname},
11935   sort={\the\glsshorttok},
11936   first={\protect\glfirstlonghyphenfont{\the\glslongtok}%
11937     \protect\glxtrfullsep{\the\glslabeltok}%
11938     \glxtrparen{\protect\glfirstabbrvhyphenfont{\the\glsshorttok}}},%
11939   firstplural={\protect\glfirstlonghyphenfont{\the\glslongpltok}%
11940     \protect\glxtrfullsep{\the\glslabeltok}%
11941     \glxtrparen{\protect\glfirstabbrvhyphenfont{\the\glsshortpltok}}},%
11942   text={\protect\glabbrvhyphenfont{\the\glsshorttok}},%
11943   plural={\protect\glabbrvhyphenfont{\the\glsshortpltok}},%
11944   description={\protect\glslonghyphenfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
11945 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11946   \glshasattribute{\the\glslabeltok}{regular}%
11947   {%
11948     \glissetattribute{\the\glslabeltok}{regular}{false}%
11949   }%
11950   {}%
11951 }%
11952 }%
11953 {%
11954   \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
11955   \renewcommand*{\glabbrvfont}[1]{\glabbrvhyphenfont{##1}}%
11956   \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvhyphenfont{##1}}%
11957   \renewcommand*{\glfirstlongfont}[1]{\glfirstlonghyphenfont{##1}}%
11958   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The first use full form and the inline form are the same for this style.

```

11959 \renewcommand*{\glxstrfullformat}[2]{%
11960   \glxstrlonghyphenshort{##1}{\glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
11961 }%
11962 \renewcommand*{\glxstrfullplformat}[2]{%
11963   \glxstrlonghyphenshort{##1}{\glsaccesslongpl{##1}}%
11964   {\glsaccessshortpl{##1}}{##2}%
11965 }%
11966 \renewcommand*{\Glsxstrfullformat}[2]{%
11967   \glxstrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
11968 }%
11969 \renewcommand*{\Glsxstrfullplformat}[2]{%
11970   \glxstrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
11971   {\glsaccessshortpl{##1}}{##2}%
11972 }%
11973 }

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

11974 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
11975 {%

```

Set accessibility attributes if enabled.

```

11976 \glxstrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

11977 \renewcommand*{\CustomAbbreviationFields}{%
11978   name={\glxstrlongshortdescname},
11979   sort={\glxstrlongshortdescsort},
11980   first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%
11981   \protect\glxstrfullsep{\the\glslabeltok}}%
11982   \glxstrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
11983   firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%
11984   \protect\glxstrfullsep{\the\glslabeltok}}%
11985   \glxstrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
11986   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
11987   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
11988 }%

```

Unset the regular attribute if it has been set.

```

11989 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11990   \glshasattribute{\the\glslabeltok}{regular}%
11991   {%
11992     \glissetattribute{\the\glslabeltok}{regular}{false}%
11993   }%
11994   {}%
11995 }%
11996 }%
11997 {%
11998   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
11999 }

```

nghyphennoshort

`\glxstrlonghyphennoshort{<label>}{<long>}{<insert>}`

```
12000 \newcommand*{\glxstrlonghyphennoshort}[3]{%
```

Grouping is needed to localise the redefinitions.

```
12001  {%
```

If *<insert>* starts with a hyphen, redefine `\glxstrwordsep` 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 `\glxstrwordsep` if *<insert>* doesn't start with a hyphen.

```
12002   \glxstrifhyphenstart{#3}{\def\glxstrwordsep{-}}{}%
```

```
12003   \glsfirstlonghyphenfont{#2\ifglxstrinsertinside{#3}\fi}%
```

```
12004   \ifglxstrinsertinside\else{#3}\fi
```

```
12005 }%
```

```
12006 }
```

short-desc-noreg This version doesn't show the short form (except explicitly with `\glxstrshort`). Since `\glxstrshort` 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. No accessibility attributes need to be set.

```
12007 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
```

```
12008 {%
```

```
12009   \renewcommand*{\CustomAbbreviationFields}{%
```

```
12010     name={\glxstrlongnoshortdescname},
```

```
12011     sort={\expandonce\glxstrorglong},
```

```
12012     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
```

```
12013     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
```

```
12014     text={\protect\glslonghyphenfont{\the\glslongtok}},%
```

```
12015     plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
```

```
12016   }%
```

Unset the regular attribute if it has been set.

```
12017   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
12018     \glshasattribute{\the\glslabeltok}{regular}%
```

```
12019     {%
```

```
12020       \glissetattribute{\the\glslabeltok}{regular}{false}%
```

```
12021     }%
```

```
12022     {}%
```

```
12023   }%
```

```
12024 }%
```

```
12025 {%
```

```
12026   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
```

In case the user wants to mix and match font styles, these are redefined here.

```
12027   \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
```

```
12028   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
```

```
12029   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
```

```

12030 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
12031 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

12032 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
12033   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
12034 }%
12035 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
12036   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
12037 }%
12038 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
12039   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
12040 }%
12041 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
12042   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
12043 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

12044 \renewcommand*{\glsxtrinlinefullformat}[2]{%
12045   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
12046   \glsxtrfullsep{##1}%
12047   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
12048 }%
12049 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
12050   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
12051   \glsxtrfullsep{##1}%
12052   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
12053 }%
12054 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
12055   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
12056   \glsxtrfullsep{##1}%
12057   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
12058 }%
12059 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
12060   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
12061   \glsxtrfullsep{##1}%
12062   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
12063 }%

```

The first use full form only displays the long form.

```

12064 \renewcommand*{\glsxtrfullformat}[2]{%
12065   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
12066 }%
12067 \renewcommand*{\glsxtrfullplformat}[2]{%
12068   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
12069 }%
12070 \renewcommand*{\Glsxtrfullformat}[2]{%
12071   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
12072 }%
12073 \renewcommand*{\Glsxtrfullplformat}[2]{%
12074   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%

```

```
12075 }%
12076 }
```

n-noshort-noreg It doesn't really make a great deal of sense to have a long-only style that doesn't have a description (unless no glossary is required), but the best course of action here is to use the short form as the name and the long form as the description.

```
12077 \newabbreviationstyle{long-hyphen-noshort-noreg}%
12078 {%
```

Set accessibility attributes if enabled.

```
12079 \glstrAccSuppAbbrSetNameShortAttrs\glscategorylabel
```

Setup the default fields.

```
12080 \renewcommand*{\CustomAbbreviationFields}{%
12081   name={\glstrlongnoshortname},
12082   sort={\the\glsshorttok},
12083   first={\protect\glstrlonghyphenfont{\the\glslongtok}},%
12084   firstplural={\protect\glstrlonghyphenfont{\the\glslongpltok}},%
12085   text={\protect\glstrlonghyphenfont{\the\glslongtok}},%
12086   plural={\protect\glstrlonghyphenfont{\the\glslongpltok}},%
12087   description={\the\glslongtok}%
12088 }%
```

Unset the regular attribute if it has been set.

```
12089 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12090   \glshasattribute{\the\glslabeltok}{regular}%
12091   {%
12092     \glissetattribute{\the\glslabeltok}{regular}{false}%
12093   }%
12094   {}%
12095 }%
12096 }%
12097 {%

12098 \GlsXtrUseAbbrStyleFmts{long-hyphen-noshort-desc-noreg}%
12099 }
```

lsxtrlonghyphen

```
\glstrlonghyphen{<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.

```
12100 \newcommand*{\glstrlonghyphen}[3]{%
```

Grouping is needed to localise the redefinitions.

```
12101 {%
12102   \glstrifhyphenstart{#3}{\def\glstrwordsep{-}}}%
12103   \glstrlonghyphenfont{#1}%
```

```
12104 }%
12105 }
```

posthyphenshort

```
\glxtrposthyphenshort{<label>}{<insert>}
```

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like `\glxtrlonghyphenshort` but omits the *<long>* part. This always uses the singular short form.

```
12106 \newcommand*{\glxtrposthyphenshort}[2]{%
12107   {%
12108     \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
12109     \ifglxtrininsertinside{\glsfirslonghyphenfont{#2}}\else{#2}\fi
12110     \glxtrfullsep{#1}%
12111     \glxtrparen
12112     {\glsfirstabbrvhyphenfont{\glstryshort{#1}}\ifglxtrininsertinside{#2}\fi}%
12113     \ifglxtrininsertinside\else{#2}\fi
12114   }%
12115 }%
12116 }
```

phensubsequent

```
\glxtrposthyphensubsequent{<label>}{<insert>}
```

Format in the post-link hook for subsequent use. The label is ignored by default.

```
12117 \newcommand*{\glxtrposthyphensubsequent}[2]{%
12118   \glxtrfont{\ifglxtrininsertinside {#2}\fi}%
12119   \ifglxtrininsertinside \else{#2}\fi
12120 }
```

ostshort-hyphen Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```
12121 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
12122 {%
```

Set accessibility attributes if enabled.

```
12123 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
12124 \renewcommand*{\CustomAbbreviationFields}{%
12125   name={\glxtrlongshortname},
12126   sort={\the\glsshorttok},
12127   first={\protect\glsfirslonghyphenfont{\the\glslongtok}},%
12128   firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}},%
12129   text={\protect\glxtrabbrvhyphenfont{\the\glsshorttok}},%
```

```

12130 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
12131 description={\protect\glslonghyphenfont{\the\glslongtok}}}%
12132 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12133 \csdef{glsxtrpostlink\glscategorylabel}{%
12134 \glsxtrifwasfirstuse
12135 {%
12136 \glsxtrposthyphenshort{\glslabel}{\glsinsert}}%
12137 }%
12138 {%

```

Put the insertion into the post-link:

```

12139 \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}}%
12140 }%
12141 }%
12142 \glsattribute{\the\glslabeltok}{regular}%
12143 {%
12144 \glssetattribute{\the\glslabeltok}{regular}{false}%
12145 }%
12146 {}%
12147 }%
12148 }%
12149 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

12150 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
12151 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
12152 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
12153 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
12154 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

12155 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
12156 \glsabbrvfont{\glsaccessshort{##1}}}%
12157 }%
12158 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
12159 \glsabbrvfont{\glsaccessshortpl{##1}}}%
12160 }%
12161 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
12162 \glsabbrvfont{\Glsaccessshort{##1}}}%
12163 }%
12164 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
12165 \glsabbrvfont{\Glsaccessshortpl{##1}}}%
12166 }%

```

First use full form:

```

12167 \renewcommand*{\glsxtrfullformat}[2]{%
12168 \glsxtrlonghyphen{\glsaccesslong{##1}}{##1}{##2}%
12169 }%
12170 \renewcommand*{\glsxtrfullplformat}[2]{%
12171 \glsxtrlonghyphen{\glsaccesslongpl{##1}}{##1}{##2}%
12172 }%

```



```

12173 \renewcommand*{\Glsxtrfullformat}[2]{%
12174   \glsxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
12175 }%
12176 \renewcommand*{\Glsxtrfullplformat}[2]{%
12177   \glsxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
12178 }%

```

In-line format.

```

12179 \renewcommand*{\glsxtrinlinefullformat}[2]{%
12180   \glsfirstlonghyphenfont{\glsaccesslong{##1}%
12181     \ifglsxtrininsertinside{##2}\fi}%
12182   \ifglsxtrininsertinside \else{##2}\fi
12183 }%
12184 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
12185   \glsfirstlonghyphenfont{\glsaccesslongpl{##1}%
12186     \ifglsxtrininsertinside{##2}\fi}%
12187   \ifglsxtrininsertinside \else{##2}\fi
12188 }%
12189 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
12190   \glsfirstlonghyphenfont{\Glsaccesslong{##1}%
12191     \ifglsxtrininsertinside{##2}\fi}%
12192   \ifglsxtrininsertinside \else{##2}\fi
12193 }%
12194 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
12195   \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
12196     \ifglsxtrininsertinside{##2}\fi}%
12197   \ifglsxtrininsertinside \else{##2}\fi
12198 }%
12199 }

```

ort-hyphen-desc Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

12200 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
12201 {%

```

Set accessibility attributes if enabled.

```

12202 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

12203 \renewcommand*{\CustomAbbreviationFields}{%
12204   name={\glsxtrlongshortdescname},
12205   sort={\glsxtrlongshortdescsort},%
12206   first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
12207   firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
12208   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
12209   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
12210 }%
12211 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12212   \csdef{glsxtrpostlink\glscategorylabel}{%
12213     \glsxtrifwasfirstuse
12214     {%

```

```

12215     \glxtrposthyphenshort{\glslabel}{\glsinsert}%
12216     }%
12217     {%
    Put the insertion into the post-link:
12218     \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
12219     }%
12220     }%
12221     \glshasattribute{\the\glslabeltok}{regular}%
12222     {%
12223     \glssetattribute{\the\glslabeltok}{regular}{false}%
12224     }%
12225     {}%
12226     }%
12227 }%
12228 {%
12229 \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
12230 }

```

shorthyphenlong

```
\glxtrshorthyphenlong{<label>}{<short>}{<long>}{<insert>}
```

The *<long>* and *<short>* arguments may be the plural form. The *<long>* argument may also be the first letter uppercase form.

```
12231 \newcommand*{\glxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
12232  {%
```

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.)

```

12233 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{}%
12234 \glsfirstabbrvhyphenfont{#2\ifglxtrininsertinside{#4}\fi}%
12235 \ifglxtrininsertinside\else{#4}\fi
12236 \glxtrfullsep{#1}%
12237 \glxtrparen{\glsfirstlonghyphenfont{#3\ifglxtrininsertinside{#4}\fi}%
12238 \ifglxtrininsertinside\else{#4}\fi}%
12239 }%
12240 }

```

hen-long-hyphen Designed for use with the `markwords` attribute.

```

12241 \newabbreviationstyle{short-hyphen-long-hyphen}%
12242 {%

```

Set accessibility attributes if enabled.

```
12243 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```

12244 \renewcommand*{\CustomAbbreviationFields}{%
12245     name={\glxstrshortlongname},
12246     sort={\the\glsshorttok},
12247     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
12248         \protect\glxstrfullsep{\the\glslabeltok}%
12249         \glxstrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
12250     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
12251         \protect\glxstrfullsep{\the\glslabeltok}%
12252         \glxstrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
12253     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
12254     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
12255     description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

12256 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12257     \glshasattribute{\the\glslabeltok}{regular}%
12258     {%
12259         \glissetattribute{\the\glslabeltok}{regular}{false}%
12260     }%
12261     {}%
12262 }%
12263 }%
12264 {%
12265     \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
12266     \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
12267     \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
12268     \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
12269     \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

12270 \renewcommand*{\glxstrfullformat}[2]{%
12271     \glxstrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
12272 }%
12273 \renewcommand*{\glxstrfullplformat}[2]{%
12274     \glxstrshorthyphenlong{##1}%
12275     {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
12276 }%
12277 \renewcommand*{\Glsxtrfullformat}[2]{%
12278     \glxstrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
12279 }%
12280 \renewcommand*{\Glsxtrfullplformat}[2]{%
12281     \glxstrshorthyphenlong{##1}%
12282     {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
12283 }%
12284 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

12285 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
12286 {%

```

Set accessibility attributes if enabled.

```
12287 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```
12288 \renewcommand*{\CustomAbbreviationFields}{%
12289   name={\glxtrshortlongdescname},
12290   sort={\glxtrshortlongdescsort},
12291   first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
12292     \protect\glxtrfullsep{\the\glslabeltok}%
12293     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
12294   firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
12295     \protect\glxtrfullsep{\the\glslabeltok}%
12296     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
12297   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
12298   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
12299 }%
```

Unset the regular attribute if it has been set.

```
12300 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12301   \glshasattribute{\the\glslabeltok}{regular}%
12302   {%
12303     \glissetattribute{\the\glslabeltok}{regular}{false}%
12304   }%
12305   {}%
12306 }%
12307 }%
12308 {%
12309   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
12310 }
```

sxtrshorthyphen

`\glxtrshorthyphen{<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.

```
12311 \newcommand*{\glxtrshorthyphen}[3]{%
```

Grouping is needed to localise the redefinitions.

```
12312 {%
12313   \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}}%
12314   \glsfirstabbrvhyphenfont{#1}%
12315 }%
12316 }
```

rposthyphenlong

```
\glxtrposthyphenlong{<label>}{<insert>}
```

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glxtrshorthyphenlong` but omits the *<short>* part. This always uses the singular long form.

```
12317 \newcommand*{\glxtrposthyphenlong}[2]{%
12318   {%
12319     \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}}%
12320     \ifglxtrininsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
12321     \glxtrfullsep{#1}%
12322     \glxtrparen
12323     {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglxtrininsertinside{#2}\fi}%
12324     \ifglxtrininsertinside\else{#2}\fi
12325   }%
12326 }%
12327 }
```

postlong-hyphen Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```
12328 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
12329 {%
```

Set accessibility attributes if enabled.

```
12330   \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
12331   \renewcommand*{\CustomAbbreviationFields}{%
12332     name={\glxtrshortlongname},
12333     sort={\the\glsshorttok},
12334     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
12335     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
12336     text={\protect\glssabbrvhyphenfont{\the\glsshorttok}},%
12337     plural={\protect\glssabbrvhyphenfont{\the\glsshortpltok}},%
12338     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
12339   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12340     \csdef{glxtrpostlink\glscategorylabel}{%
12341       \glxtrifwasfirstuse
12342       {%
12343         \glxtrposthyphenlong{\glslabel}{\glsininsert}%
12344       }%
12345     }%
```

Put the insertion into the post-link:

```
12346       \glxtrposthyphensubsequent{\glslabel}{\glsininsert}%
12347     }%
12348   }%
12349   \glshasattribute{\the\glslabeltok}{regular}%
12350   {%
12351     \glissetattribute{\the\glslabeltok}{regular}{false}%
12352   }%
12353 }
```

```

12354 }%
12355 }%
12356 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

12357 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
12358 \renewcommand*\glsabbrvfont[1]{\glsabbrvhyphenfont{##1}}%
12359 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvhyphenfont{##1}}%
12360 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonghyphenfont{##1}}%
12361 \renewcommand*\glslongfont[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

12362 \renewcommand*\glxtrsubsequentfmt[2]{%
12363   \glsabbrvfont{\glsaccessshort{##1}}%
12364 }%
12365 \renewcommand*\glxtrsubsequentplfmt[2]{%
12366   \glsabbrvfont{\glsaccessshortpl{##1}}%
12367 }%
12368 \renewcommand*\Glsxtrsubsequentfmt[2]{%
12369   \glsabbrvfont{\Glsaccessshort{##1}}%
12370 }%
12371 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
12372   \glsabbrvfont{\Glsaccessshortpl{##1}}%
12373 }%

```

First use full form:

```

12374 \renewcommand*\glxtrfullformat[2]{%
12375   \glxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
12376 }%
12377 \renewcommand*\glxtrfullplformat[2]{%
12378   \glxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
12379 }%
12380 \renewcommand*\Glsxtrfullformat[2]{%
12381   \glxtrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
12382 }%
12383 \renewcommand*\Glsxtrfullplformat[2]{%
12384   \glxtrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
12385 }%

```

In-line format. Commands like `\glxtrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

12386 \renewcommand*\glxtrinlinefullformat[2]{%
12387   \glsfirstabbrvhyphenfont{\glsaccessshort{##1}}%
12388   \ifglxtrininsertinside{##2}\fi}%
12389   \ifglxtrininsertinside \else{##2}\fi
12390 }%
12391 \renewcommand*\glxtrinlinefullplformat[2]{%
12392   \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}}%
12393   \ifglxtrininsertinside{##2}\fi}%
12394   \ifglxtrininsertinside \else{##2}\fi
12395 }%

```

```

12396 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
12397   \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}}%
12398   \ifglxtrinsertinside{##2}\fi}%
12399   \ifglxtrinsertinside \else{##2}\fi
12400 }%
12401 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
12402   \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}}%
12403   \ifglxtrinsertinside{##2}\fi}%
12404   \ifglxtrinsertinside \else{##2}\fi
12405 }%
12406 }

```

ong-hyphen-desc Like short-hyphen-postlong-hyphen but the description must be supplied by the user.

```

12407 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
12408 {%

```

Set accessibility attributes if enabled.

```

12409 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

12410 \renewcommand*{\CustomAbbreviationFields}{%
12411   name={\glxtrshortlongdescname},
12412   sort={\glxtrshortlongdescsort},%
12413   first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
12414   firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
12415   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
12416   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
12417 }%
12418 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12419   \csdef{glxtrpostlink\glscategorylabel}{%
12420     \glxtrifwasfirstuse
12421     {%
12422       \glxtrposthyphenlong{\glslabel}{\glsinsert}%
12423     }%
12424     {%

```

Put the insertion into the post-link:

```

12425       \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
12426     }%
12427   }%
12428   \glshasattribute{\the\glslabeltok}{regular}%
12429   {%
12430     \glissetattribute{\the\glslabeltok}{regular}{false}%
12431   }%
12432   {}%
12433 }%
12434 }%
12435 {%
12436   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
12437 }

```

1.7.8 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

```
12438 \newcommand*{\glsabbrvonlyfont}{\glsabbrvdefaultfont}%
```

stabbrvonlyfont

```
12439 \newcommand*{\glsfirstabbrvonlyfont}{\glsabbrvonlyfont}%
```

glslongonlyfont

```
12440 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%
```

rstlongonlyfont

```
12441 \newcommand*{\glsfirstlongonlyfont}{\glslongonlyfont}%
```

The default short form suffix:

lsxtronlysuffix

```
12442 \newcommand*{\glsxtronlysuffix}{\glsxtrabbrvpluralsuffix}
```

\glsxtronlyname The default name format for this style.

```
12443 \newcommand*{\glsxtronlyname}{%
```

```
12444   \protect\glsabbrvonlyfont{\the\glsshorttok}}%
```

```
12445 }
```

only-short-only

```
12446 \newabbreviationstyle{long-only-short-only}%
```

```
12447 {%
```

Set accessibility attributes if enabled.

```
12448   \glsxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
12449   \renewcommand*{\CustomAbbreviationFields}{%
```

```
12450     name={\glsxtronlyname},
```

```
12451     sort={\the\glsshorttok},
```

```
12452     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
```

```
12453     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
```

```
12454     text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%
```

```
12455     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
```

```
12456     description={\protect\glslongonlyfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
12457   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
12458     \glshasattribute{\the\glslabeltok}{regular}%
```

```
12459     {%
```

```
12460       \glssetattribute{\the\glslabeltok}{regular}{false}%
```

```
12461     }%
```

```
12462   }%
```



```

12463 }%
12464 }%
12465 {%

```

```

12466 \renewcommand*{\abbrvpluralsuffix}{\glstronlysuffix}%
12467 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvonlyfont{##1}}%
12468 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
12469 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
12470 \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%

```

The first use full form doesn't show the short form.

```

12471 \renewcommand*{\glstrfullformat}[2]{%
12472   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglstrinsertinside##2\fi}%
12473   \ifglstrinsertinside\else##2\fi
12474 }%
12475 \renewcommand*{\glstrfullplformat}[2]{%
12476   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
12477   \ifglstrinsertinside\else##2\fi
12478 }%
12479 \renewcommand*{\Glsstrfullformat}[2]{%
12480   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglstrinsertinside##2\fi}%
12481   \ifglstrinsertinside\else##2\fi
12482 }%
12483 \renewcommand*{\Glsstrfullplformat}[2]{%
12484   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
12485   \ifglstrinsertinside\else##2\fi
12486 }%

```

The inline full form does show the short form.

```

12487 \renewcommand*{\glstrinlinefullformat}[2]{%
12488   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglstrinsertinside##2\fi}%
12489   \ifglstrinsertinside\else##2\fi
12490   \glstrfullsep{##1}%
12491   \glstrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
12492 }%
12493 \renewcommand*{\glstrinlinefullplformat}[2]{%
12494   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
12495   \ifglstrinsertinside\else##2\fi
12496   \glstrfullsep{##1}%
12497   \glstrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
12498 }%
12499 \renewcommand*{\Glsstrinlinefullformat}[2]{%
12500   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglstrinsertinside##2\fi}%
12501   \ifglstrinsertinside\else##2\fi
12502   \glstrfullsep{##1}%
12503   \glstrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
12504 }%
12505 \renewcommand*{\Glsstrinlinefullplformat}[2]{%
12506   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
12507   \ifglstrinsertinside\else##2\fi

```

```

12508     \glxtrfullsep{##1}%
12509     \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
12510 }%
12511 }

```

xtronlydescsort

```

12512 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

12513 \newcommand*{\glxtronlydescname}{%
12514   \protect\glslongfont{\the\glslongtok}%
12515 }

```

short-only-desc

```

12516 \newabbreviationstyle{long-only-short-only-desc}%
12517 {%

```

Set accessibility attributes if enabled.

```

12518   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

12519   \renewcommand*{\CustomAbbreviationFields}{%
12520     name={\glxtronlydescname},
12521     sort={\glxtronlydescsort},%
12522     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
12523     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
12524     text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%
12525     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}}}%
12526 }%

```

Unset the regular attribute if it has been set.

```

12527   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12528     \glshasattribute{\the\glslabeltok}{regular}%
12529     {%
12530       \glissetattribute{\the\glslabeltok}{regular}{false}%
12531     }%
12532   }%
12533 }%
12534 }%
12535 {%
12536   \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
12537 }

```

Small-caps is awkward, so support for that is added.

abbrvsconlyfont

```

12538 \newcommand*{\glsabbrvsconlyfont}{\glsabbrvscfont}%

```

abbrvsconlyfont

```

12539 \newcommand*{\glsfirstabbrvsconlyfont}{\glsabbrvsconlyfont}%

```

The default short form suffix:

xlstrsconlysuffix

```
12540 \newcommand*{\glxtrscnlysuffix}{\glxtrscsuffix}
```

xlstrsconlyname The default name format for this style.

```
12541 \newcommand*{\glxtrscnlyname}{%
12542   \protect\glabbrvsconlyfont{\the\glsshorttok}}%
12543 }
```

y-short-sc-only

```
12544 \newabbreviationstyle{long-only-short-sc-only}%
12545 {%
```

Set accessibility attributes if enabled.

```
12546   \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
12547   \renewcommand*{\CustomAbbreviationFields}{%
12548     name={\glxtrscnlyname},
12549     sort={\the\glsshorttok},
12550     first={\protect\glfirstlongonlyfont{\the\glslongtok}},%
12551     firstplural={\protect\glfirstlongonlyfont{\the\glslongpltok}},%
12552     text={\protect\glabbrvsconlyfont{\the\glsshorttok}},%
12553     plural={\protect\glabbrvsconlyfont{\the\glsshortpltok}},%
12554     description={\protect\glslongonlyfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
12555   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12556     \glshasattribute{\the\glslabeltok}{regular}}%
12557   {%
12558     \glissetattribute{\the\glslabeltok}{regular}{false}}%
12559   }%
12560   {}%
12561   }%
12562 }%
12563 {%
12564   \renewcommand*{\abbrvpluralsuffix}{\glxtrscnlysuffix}%
12565   \renewcommand*{\glabbrvfont}[1]{\glabbrvsconlyfont{##1}}%
12566   \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvsconlyfont{##1}}%
12567   \renewcommand*{\glfirstlongfont}[1]{\glfirstlongonlyfont{##1}}%
12568   \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%
```

The first use full form doesn't show the short form.

```
12569   \renewcommand*{\glxtrfullformat}[2]{%
12570     \glfirstlongonlyfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
12571     \ifglxtrininsertinside\else##2\fi
12572   }%
12573   \renewcommand*{\glxtrfullplformat}[2]{%
12574     \glfirstlongonlyfont{\glaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
12575     \ifglxtrininsertinside\else##2\fi
```

```

12576 }%
12577 \renewcommand*{\Glsxtrfullformat}[2]{%
12578   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
12579   \ifglxtrinsertinside\else##2\fi
12580 }%
12581 \renewcommand*{\Glsxtrfullplformat}[2]{%
12582   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
12583   \ifglxtrinsertinside\else##2\fi
12584 }%

```

The inline full form does show the short form.

```

12585 \renewcommand*{\glxtrinlinefullformat}[2]{%
12586   \glsfirstlongonlyfont{\glaccesslong{##1}\ifglxtrinsertinside##2\fi}%
12587   \ifglxtrinsertinside\else##2\fi
12588   \glxtrfullsep{##1}%
12589   \glxtrparen{\protect\glfirstabbrvsonlyfont{\glaccessshort{##1}}}%
12590 }%
12591 \renewcommand*{\glxtrinlinefullplformat}[2]{%
12592   \glsfirstlongonlyfont{\glaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
12593   \ifglxtrinsertinside\else##2\fi
12594   \glxtrfullsep{##1}%
12595   \glxtrparen{\protect\glfirstabbrvsonlyfont{\glaccessshortpl{##1}}}%
12596 }%
12597 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
12598   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
12599   \ifglxtrinsertinside\else##2\fi
12600   \glxtrfullsep{##1}%
12601   \glxtrparen{\protect\glfirstabbrvsonlyfont{\glaccessshortpl{##1}}}%
12602 }%
12603 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
12604   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
12605   \ifglxtrinsertinside\else##2\fi
12606   \glxtrfullsep{##1}%
12607   \glxtrparen{\protect\glfirstabbrvsonlyfont{\Glsaccessshortpl{##1}}}%
12608 }%
12609 }

```

rsconlydescsort

```
12610 \newcommand*{\glxtrsconlydescsort}{\glxtronlydescsort}
```

rsconlydescname

```
12611 \newcommand*{\glxtrsconlydescname}{\glxtronlydescname}
```

rt-sc-only-desc

```

12612 \newabbreviationstyle{long-only-short-sc-only-desc}%
12613 {}

```

Set accessibility attributes if enabled.

```
12614 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```
12615 \renewcommand*{\CustomAbbreviationFields}{%
12616   name={\glstrsconlydescname},
12617   sort={\glstrsconlydescsort},%
12618   first={\protect\glfirstlongonlyfont{\the\glslongtok}},%
12619   firstplural={\protect\glfirstlongonlyfont{\the\glslongpltok}},%
12620   text={\protect\glabbrvsconlyfont{\the\glsshorttok}},%
12621   plural={\protect\glabbrvsconlyfont{\the\glsshortpltok}}%
12622 }%
```

Unset the regular attribute if it has been set.

```
12623 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12624   \glshasattribute{\the\glslabeltok}{regular}%
12625   {%
12626     \glissetattribute{\the\glslabeltok}{regular}{false}%
12627   }%
12628   {}%
12629 }%
12630 }%
12631 {%
12632   \GlsXtrUseAbbrStyleFmts{long-only-short-sc-only}%
12633 }
```

1.8 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 `\glstryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glabbrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref`'s `\texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glstrshort` 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:

```
12634 \let\@glxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
12635 \renewcommand*{\markright}[1]{%
```

```
12636   \glxtrmarkhook
```

```
12637   \@glxtr@org@markright{\@glxtrinmark#1\@glxtrnotinmark}%
```

```
12638   \glxtrrestoremarkhook
```

```
12639 }
```

`\markboth` Save original definition:

```
12640 \let\@glxtr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
12641 \renewcommand*{\markboth}[2]{%
```

```
12642   \glxtrmarkhook
```

```
12643   \@glxtr@org@markboth
```

```
12644     {\@glxtrinmark#1\@glxtrnotinmark}%
```

```
12645     {\@glxtrinmark#2\@glxtrnotinmark}%
```

```
12646   \glxtrrestoremarkhook
```

```
12647 }
```

Also do this for `\@starttoc`

`\@starttoc` Save original definition:

```
12648 \let\@glxtr@org@@starttoc\@starttoc
```

Redefine:

```
12649 \renewcommand*{\@starttoc}[1]{%
```

```
12650   \glxtrmarkhook
```

```
12651   \@glxtrinmark
```

```
12652   \@glxtr@org@@starttoc{#1}%
```

```
12653   \@glxtrnotinmark
```

```
12654   \glxtrrestoremarkhook
```

```
12655 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

`sxtrRevertMarks`

```
12656 \newcommand*{\glxtrRevertMarks}{%
```

```
12657   \let\markright\@glxtr@org@markright
```

```
12658   \let\markboth\@glxtr@org@markboth
```

```
12659   \let\@starttoc\@glxtr@org@@starttoc
```

```
12660 }
```

`rRevertTocMarks` Just restores `\@starttoc`.

```
12661 \newcommand*{\glxtrRevertTocMarks}{%
```

```
12662   \let\@starttoc\@glxtr@org@@starttoc
```

```
12663 }
```

`\glxtrifinmark`

```
12664 \newcommand*{\glxtrifinmark}[2]{#2}
```

`\@glxtrinmark`

```
12665 \newrobustcmd*{\@glxtrinmark}{%
12666   \let\glxtrifinmark\@firstoftwo
12667 }
```

`glxstrnotinmark`

```
12668 \newrobustcmd*{\glxstrnotinmark}{%
12669   \let\glxtrifinmark\@secondoftwo
12670 }
```

`teorpdforheading`

```
12671 \ifdef\texorpdfstring
12672 {
12673   \newcommand*{\glxstrtitleorpdforheading}[3]{\texorpdfstring{#1}{#2}}
12674 }
12675 {
12676   \newcommand*{\glxstrtitleorpdforheading}[3]{#1}
12677 }
```

`\glxstrmarkhook` Hook used in new definition of `\markboth` and `\markright` to make some changes to apply to the marks:

```
12678 \newcommand*{\glxstrmarkhook}{%
```

Save current definitions:

```
12679 \let\@glxstr@org@MakeUppercase\MakeUppercase
12680 \let\@glxstr@org@glxstrtitleorpdforheading\glxstrtitleorpdforheading
12681 \let\@glxstr@org@glxstrtitleshort\glxstrtitleshort
12682 \let\@glxstr@org@glxstrtitleshortpl\glxstrtitleshortpl
12683 \let\@glxstr@org@Glsxtrtitleshort\Glsxtrtitleshort
12684 \let\@glxstr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
12685 \let\@glxstr@org@glxstrtitlename\glxstrtitlename
12686 \let\@glxstr@org@Glsxtrtitlename\Glsxtrtitlename
12687 \let\@glxstr@org@glxstrtitletext\glxstrtitletext
12688 \let\@glxstr@org@Glsxtrtitletext\Glsxtrtitletext
12689 \let\@glxstr@org@glxstrtitleplural\glxstrtitleplural
12690 \let\@glxstr@org@Glsxtrtitleplural\Glsxtrtitleplural
12691 \let\@glxstr@org@glxstrtitlefirst\glxstrtitlefirst
12692 \let\@glxstr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
12693 \let\@glxstr@org@glxstrtitlefirstplural\glxstrtitlefirstplural
12694 \let\@glxstr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
12695 \let\@glxstr@org@glxstrtitlelong\glxstrtitlelong
12696 \let\@glxstr@org@glxstrtitlelongpl\glxstrtitlelongpl
12697 \let\@glxstr@org@Glsxtrtitlelong\Glsxtrtitlelong
12698 \let\@glxstr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
12699 \let\@glxstr@org@glxstrtitlefull\glxstrtitlefull
12700 \let\@glxstr@org@glxstrtitlefullpl\glxstrtitlefullpl
```

```

12701 \let\@glxstr@org@Glsxtrtitlefull\Glsxtrtitlefull
12702 \let\@glxstr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl

```

New definitions

```

12703 \let\glxstrifinmark\@firstoftwo
12704 \let\MakeUppercase\MakeTextUppercase
12705 \let\glxstrtitleorpdforheading\@thirdofthree
12706 \let\glxstrtitleshort\glxstrheadshort
12707 \let\glxstrtitleshortpl\glxstrheadshortpl
12708 \let\Glsxstrtitleshort\Glsxtrheadshort
12709 \let\Glsxstrtitleshortpl\Glsxtrheadshortpl
12710 \let\glxstrtitlename\glxstrheadname
12711 \let\Glsxstrtitlename\Glsxtrheadname
12712 \let\glxstrtitletext\glxstrheadtext
12713 \let\Glsxstrtitletext\Glsxtrheadtext
12714 \let\glxstrtitleplural\glxstrheadplural
12715 \let\Glsxstrtitleplural\Glsxtrheadplural
12716 \let\glxstrtitlefirst\glxstrheadfirst
12717 \let\Glsxstrtitlefirst\Glsxtrheadfirst
12718 \let\glxstrtitlefirstplural\glxstrheadfirstplural
12719 \let\Glsxstrtitlefirstplural\Glsxtrheadfirstplural
12720 \let\glxstrtitlelong\glxstrheadlong
12721 \let\glxstrtitlelongpl\glxstrheadlongpl
12722 \let\Glsxstrtitlelong\Glsxtrheadlong
12723 \let\Glsxstrtitlelongpl\Glsxtrheadlongpl
12724 \let\glxstrtitlefull\glxstrheadfull
12725 \let\glxstrtitlefullpl\glxstrheadfullpl
12726 \let\Glsxstrtitlefull\Glsxtrheadfull
12727 \let\Glsxstrtitlefullpl\Glsxtrheadfullpl
12728 }

```

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.)

```

12729 \newcommand*{\glxstrrestoremarkhook}{%
12730 \let\glxstrifinmark\@secondoftwo
12731 \let\MakeUppercase\@glxstr@org@MakeUppercase
12732 \let\glxstrtitleorpdforheading\@glxstr@org@glxstrtitleorpdforheading
12733 \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
12734 \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
12735 \let\Glsxstrtitleshort\@glxstr@org@Glsxstrtitleshort
12736 \let\Glsxstrtitleshortpl\@glxstr@org@Glsxstrtitleshortpl
12737 \let\glxstrtitlename\@glxstr@org@glxstrtitlename
12738 \let\Glsxstrtitlename\@glxstr@org@Glsxstrtitlename
12739 \let\glxstrtitletext\@glxstr@org@glxstrtitletext
12740 \let\Glsxstrtitletext\@glxstr@org@Glsxstrtitletext
12741 \let\glxstrtitleplural\@glxstr@org@glxstrtitleplural
12742 \let\Glsxstrtitleplural\@glxstr@org@Glsxstrtitleplural

```



```

12743 \let\glxstrtitlefirst\@glxstr@org@glxstrtitlefirst
12744 \let\Glsxstrtitlefirst\@glxstr@org@Glsxstrtitlefirst
12745 \let\glxstrtitlefirstplural\@glxstr@org@glxstrtitlefirstplural
12746 \let\Glsxstrtitlefirstplural\@glxstr@org@Glsxstrtitlefirstplural
12747 \let\glxstrtitlelong\@glxstr@org@glxstrtitlelong
12748 \let\glxstrtitlelongpl\@glxstr@org@glxstrtitlelongpl
12749 \let\Glsxstrtitlelong\@glxstr@org@Glsxstrtitlelong
12750 \let\Glsxstrtitlelongpl\@glxstr@org@Glsxstrtitlelongpl
12751 \let\glxstrtitlefull\@glxstr@org@glxstrtitlefull
12752 \let\glxstrtitlefullpl\@glxstr@org@glxstrtitlefullpl
12753 \let\Glsxstrtitlefull\@glxstr@org@Glsxstrtitlefull
12754 \let\Glsxstrtitlefullpl\@glxstr@org@Glsxstrtitlefullpl
12755 }

```

Instead of using one document-wide conditional, use headuc attribute to determine whether or not to use the all upper case form.

glxstrheadshort Command used to display short form in the page header.

```

12756 \newcommand*{\glxstrheadshort}[1]{%
12757 \protect\NoCaseChange
12758 {%
12759 \gl@ifattribute{#1}{headuc}{true}%
12760 {%
12761 \GLSxtrshort[noindex,hyper=false]{#1}[]%
12762 }%
12763 {%
12764 \glxstrshort[noindex,hyper=false]{#1}[]%
12765 }%
12766 }%
12767 }

```

lsxstrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

12768 \newrobustcmd*{\lsxstrtitleshort}[1]{%
12769 \glxstrshort[noindex,hyper=false]{#1}[]%
12770 }

```

sxstrheadshortpl 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.

```

12771 \newcommand*{\glxstrheadshortpl}[1]{%
12772 \protect\NoCaseChange
12773 {%
12774 \gl@ifattribute{#1}{headuc}{true}%
12775 {%
12776 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
12777 }%
12778 {%
12779 \glxstrshortpl[noindex,hyper=false]{#1}[]%
12780 }%

```

```
12781 }%
12782 }
```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```
12783 \newrobustcmd*{\glxtrtitleshortpl}[1]{%
12784   \glxtrshortpl[noindex,hyper=false]{#1}[]%
12785 }
```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```
12786 \newcommand*{\Glsxtrheadshort}[1]{%
12787   \protect\NoCaseChange
12788   {%
12789     \glsifattribute{#1}{headuc}{true}%
12790     {%
12791       \GLSxtrshort[noindex,hyper=false]{#1}[]%
12792     }%
12793     {%
12794       \Glsxtrshort[noindex,hyper=false]{#1}[]%
12795     }%
12796   }%
12797 }
```

lslxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
12798 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
12799   \Glsxtrshort[noindex,hyper=false]{#1}[]%
12800 }
```

LSxtrtitleshort Command to display short form of abbreviation in section title and table of contents in all upper case.

```
12801 \newrobustcmd*{\GLSxtrtitleshort}[1]{%
12802   \GLSxtrshort[noindex,hyper=false]{#1}[]%
12803 }
```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```
12804 \newcommand*{\Glsxtrheadshortpl}[1]{%
12805   \protect\NoCaseChange
12806   {%
12807     \glsifattribute{#1}{headuc}{true}%
12808     {%
12809       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
12810     }%
12811     {%
12812       \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
12813     }%
12814   }%
12815 }
```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
12816 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
12817   \Glsxtrshortpl [noindex,hyper=false]{#1} []%
12818 }
```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents in all upper case.

```
12819 \newrobustcmd*{\GLSxtrtitleshortpl}[1]{%
12820   \GLSxtrshortpl [noindex,hyper=false]{#1} []%
12821 }
```

`\glxtrheadname` As above but for the name value.

```
12822 \newcommand*{\glxtrheadname}[1]{%
12823   \protect\NoCaseChange
12824   {%
12825     \glsifattribute{#1}{headuc}{true}%
12826     {%
12827       \GLSname [noindex,hyper=false]{#1} []%
12828     }%
12829     {%
12830       \glsname [noindex,hyper=false]{#1} []%
12831     }%
12832   }%
12833 }
```

`glxtrtitlename` Command to display name value in section title and table of contents.

```
12834 \newrobustcmd*{\glxtrtitlename}[1]{%
12835   \glsname [noindex,hyper=false]{#1} []%
12836 }
```

`\Glsxtrheadname` First letter converted to upper case

```
12837 \newcommand*{\Glsxtrheadname}[1]{%
12838   \protect\NoCaseChange
12839   {%
12840     \glsifattribute{#1}{headuc}{true}%
12841     {%
12842       \GLSname [noindex,hyper=false]{#1} []%
12843     }%
12844     {%
12845       \Glsname [noindex,hyper=false]{#1} []%
12846     }%
12847   }%
12848 }
```

`Glsxtrtitlename` Command to display name value in section title and table of contents with the first letter changed to upper case.

```
12849 \newrobustcmd*{\Glsxtrtitlename}[1]{%
```

```

12850 \Glsname[noindex,hyper=false]{#1}[]%
12851 }

```

GLSxtrtitlename Command to display name value in section title and table of contents in all upper case.

```

12852 \newrobustcmd*{\GLSxtrtitlename}[1]{%
12853 \Glsname[noindex,hyper=false]{#1}[]%
12854 }

```

\glxtrheadtext As above but for the text value.

```

12855 \newcommand*{\glxtrheadtext}[1]{%
12856 \protect\NoCaseChange
12857 {%
12858 \glsifattribute{#1}{headuc}{true}%
12859 {%
12860 \GLStext[noindex,hyper=false]{#1}[]%
12861 }%
12862 {%
12863 \glstext[noindex,hyper=false]{#1}[]%
12864 }%
12865 }%
12866 }

```

glxtrtitletext Command to display text value in section title and table of contents.

```

12867 \newrobustcmd*{\glxtrtitletext}[1]{%
12868 \glstext[noindex,hyper=false]{#1}[]%
12869 }

```

\Glsxtrheadtext First letter converted to upper case

```

12870 \newcommand*{\Glsxtrheadtext}[1]{%
12871 \protect\NoCaseChange
12872 {%
12873 \glsifattribute{#1}{headuc}{true}%
12874 {%
12875 \GLStext[noindex,hyper=false]{#1}[]%
12876 }%
12877 {%
12878 \Glstext[noindex,hyper=false]{#1}[]%
12879 }%
12880 }%
12881 }

```

Glsxtrtitletext Command to display text value in section title and table of contents with the first letter changed to upper case.

```

12882 \newrobustcmd*{\Glsxtrtitletext}[1]{%
12883 \Glstext[noindex,hyper=false]{#1}[]%
12884 }

```

GLSxtrtitletext Command to display text value in section title and table of contents in all upper case.

```

12885 \newrobustcmd*{\GLSxtrtitletext}[1]{%
12886   \GLSxtext[noindex,hyper=false]{#1}[]%
12887 }

```

lsxtrheadplural As above but for the plural value.

```

12888 \newcommand*{\glxtrheadplural}[1]{%
12889   \protect\NoCaseChange
12890   {%
12891     \glsifattribute{#1}{headuc}{true}%
12892     {%
12893       \GLSplural[noindex,hyper=false]{#1}[]%
12894     }%
12895     {%
12896       \glsplural[noindex,hyper=false]{#1}[]%
12897     }%
12898   }%
12899 }

```

sxtrtitleplural Command to display plural value in section title and table of contents.

```

12900 \newrobustcmd*{\glxtrtitleplural}[1]{%
12901   \glsplural[noindex,hyper=false]{#1}[]%
12902 }

```

lsxtrheadplural Convert first letter to upper case.

```

12903 \newcommand*{\Glsxtrheadplural}[1]{%
12904   \protect\NoCaseChange
12905   {%
12906     \glsifattribute{#1}{headuc}{true}%
12907     {%
12908       \GLSplural[noindex,hyper=false]{#1}[]%
12909     }%
12910     {%
12911       \Glsplural[noindex,hyper=false]{#1}[]%
12912     }%
12913   }%
12914 }

```

sxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```

12915 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
12916   \Glsplural[noindex,hyper=false]{#1}[]%
12917 }

```

Sxtrtitleplural Command to display plural value in section title and table of contents in all upper case.

```

12918 \newrobustcmd*{\GLSxtrtitleplural}[1]{%
12919   \GLSplural[noindex,hyper=false]{#1}[]%
12920 }

```

glsxtrheadfirst As above but for the first value.

```
12921 \newcommand*{\glsxtrheadfirst}[1]{%
12922   \protect\NoCaseChange
12923   {%
12924     \glsifattribute{#1}{headuc}{true}%
12925     {%
12926       \GLSfirst[noindex,hyper=false]{#1}[]%
12927     }%
12928     {%
12929       \glsfirst[noindex,hyper=false]{#1}[]%
12930     }%
12931   }%
12932 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```
12933 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
12934   \glsfirst[noindex,hyper=false]{#1}[]%
12935 }
```

Glsxtrheadfirst First letter converted to upper case

```
12936 \newcommand*{\Glsxtrheadfirst}[1]{%
12937   \protect\NoCaseChange
12938   {%
12939     \glsifattribute{#1}{headuc}{true}%
12940     {%
12941       \GLSfirst[noindex,hyper=false]{#1}[]%
12942     }%
12943     {%
12944       \Glsfirst[noindex,hyper=false]{#1}[]%
12945     }%
12946   }%
12947 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```
12948 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
12949   \Glsfirst[noindex,hyper=false]{#1}[]%
12950 }
```

LSxtrtitlefirst Command to display first value in section title and table of contents in all upper case.

```
12951 \newrobustcmd*{\GLSxtrtitlefirst}[1]{%
12952   \GLSfirst[noindex,hyper=false]{#1}[]%
12953 }
```

headfirstplural As above but for the firstplural value.

```
12954 \newcommand*{\glsxtrheadfirstplural}[1]{%
12955   \protect\NoCaseChange
12956   {%
```

```

12957 \glsifattribute{#1}{headuc}{true}%
12958 {%
12959 \GLSfirstplural[noindex,hyper=false]{#1}[]%
12960 }%
12961 {%
12962 \glsfirstplural[noindex,hyper=false]{#1}[]%
12963 }%
12964 }%
12965 }

```

`\titlefirstplural` Command to display firstplural value in section title and table of contents.

```

12966 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
12967 \glsfirstplural[noindex,hyper=false]{#1}[]%
12968 }

```

`\headfirstplural` First letter converted to upper case

```

12969 \newcommand*{\Glsxtrheadfirstplural}[1]{%
12970 \protect\NoCaseChange
12971 {%
12972 \glsifattribute{#1}{headuc}{true}%
12973 {%
12974 \GLSfirstplural[noindex,hyper=false]{#1}[]%
12975 }%
12976 {%
12977 \Glsfirstplural[noindex,hyper=false]{#1}[]%
12978 }%
12979 }%
12980 }

```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```

12981 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
12982 \Glsfirstplural[noindex,hyper=false]{#1}[]%
12983 }

```

`\titlefirstplural` Command to display first value in section title and table of contents in all upper case.

```

12984 \newrobustcmd*{\GLSxtrtitlefirstplural}[1]{%
12985 \GLSfirstplural[noindex,hyper=false]{#1}[]%
12986 }

```

`\glsxtrheadlong` Command used to display long form in the page header.

```

12987 \newcommand*{\glsxtrheadlong}[1]{%
12988 \protect\NoCaseChange
12989 {%
12990 \glsifattribute{#1}{headuc}{true}%
12991 {%
12992 \GLSxtrlong[noindex,hyper=false]{#1}[]%
12993 }%

```

```

12994   {%
12995       \glxstrlong[noindex,hyper=false]{#1}[]%
12996   }%
12997 }%
12998 }

```

`\glxstrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```

12999 \newrobustcmd*{\glxstrtitlelong}[1]{%
13000     \glxstrlong[noindex,hyper=false]{#1}[]%
13001 }

```

`\lsxtrheadlongpl` Command used to display plural long form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrlongpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

13002 \newcommand*{\lsxtrheadlongpl}[1]{%
13003     \protect\NoCaseChange
13004     {%
13005         \gl@ifattribute{#1}{headuc}{true}%
13006         {%
13007             \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
13008         }%
13009         {%
13010             \glxstrlongpl[noindex,hyper=false]{#1}[]%
13011         }%
13012     }%
13013 }

```

`\sxstrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```

13014 \newrobustcmd*{\sxstrtitlelongpl}[1]{%
13015     \glxstrlongpl[noindex,hyper=false]{#1}[]%
13016 }

```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

13017 \newcommand*{\Glsxtrheadlong}[1]{%
13018     \protect\NoCaseChange
13019     {%
13020         \gl@ifattribute{#1}{headuc}{true}%
13021         {%
13022             \GLSxtrlong[noindex,hyper=false]{#1}[]%
13023         }%
13024         {%
13025             \Glsxtrlong[noindex,hyper=false]{#1}[]%
13026         }%
13027     }%
13028 }

```

`\Glsxstrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.


```

13029 \newrobustcmd*{\GLSxtrtitlelong}[1]{%
13030   \GLSxtrlong[noindex,hyper=false]{#1}[]%
13031 }

```

GLSxtrtitlelong Command to display long form of abbreviation in section title and table of contents in all upper case.

```

13032 \newrobustcmd*{\GLSxtrtitlelong}[1]{%
13033   \GLSxtrlong[noindex,hyper=false]{#1}[]%
13034 }

```

lsxtrheadlongpl Command used to display plural long form in the page header with the first letter converted to upper case.

```

13035 \newcommand*{\GLSxtrheadlongpl}[1]{%
13036   \protect\NoCaseChange
13037   {%
13038     \glsifattribute{#1}{headuc}{true}%
13039     {%
13040       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
13041     }%
13042     {%
13043       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
13044     }%
13045   }%
13046 }

```

sxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

13047 \newrobustcmd*{\GLSxtrtitlelongpl}[1]{%
13048   \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
13049 }

```

Sxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents in all upper case.

```

13050 \newrobustcmd*{\GLSxtrtitlelongpl}[1]{%
13051   \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
13052 }

```

\glsxtrheadfull Command used to display full form in the page header.

```

13053 \newcommand*{\glsxtrheadfull}[1]{%
13054   \protect\NoCaseChange
13055   {%
13056     \glsifattribute{#1}{headuc}{true}%
13057     {%
13058       \GLSxtrfull[noindex,hyper=false]{#1}[]%
13059     }%
13060     {%
13061       \GLSxtrfull[noindex,hyper=false]{#1}[]%
13062     }%

```

```

13063 }%
13064 }

```

`\glstrtitlefull` Command to display full form of abbreviation in section title and table of contents.

```

13065 \newrobustcmd*{\glstrtitlefull}[1]{%
13066   \glstrfull[noindex,hyper=false]{#1}[]%
13067 }

```

`\lsxtrheadfullpl` Command used to display plural full form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrfullpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

13068 \newcommand*{\lsxtrheadfullpl}[1]{%
13069   \protect\NoCaseChange
13070   {%
13071     \gl@ifattribute{#1}{headuc}{true}%
13072     {%
13073       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
13074     }%
13075     {%
13076       \glstrfullpl[noindex,hyper=false]{#1}[]%
13077     }%
13078   }%
13079 }

```

`\sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents.

```

13080 \newrobustcmd*{\sxtrtitlefullpl}[1]{%
13081   \glstrfullpl[noindex,hyper=false]{#1}[]%
13082 }

```

`\Glsxtrheadfull` Command used to display full form in the page header with the first letter converted to upper case.

```

13083 \newcommand*{\Glsxtrheadfull}[1]{%
13084   \protect\NoCaseChange
13085   {%
13086     \gl@ifattribute{#1}{headuc}{true}%
13087     {%
13088       \GLSxtrfull[noindex,hyper=false]{#1}[]%
13089     }%
13090     {%
13091       \Glsxtrfull[noindex,hyper=false]{#1}[]%
13092     }%
13093   }%
13094 }

```

`\Glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

13095 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
13096   \Glsxtrfull[noindex,hyper=false]{#1}[]%
13097 }

```

GLSxtrtitlefull Command to display full form of abbreviation in section title and table of contents in all upper case.

```
13098 \newrobustcmd*{\GLSxtrtitlefull}[1]{%
13099   \GLSxtrfull[noindex,hyper=false]{#1}[]%
13100 }
```

lsxtrheadfullpl Command used to display plural full form in the page header with the first letter converted to upper case.

```
13101 \newcommand*{\Glsxtrheadfullpl}[1]{%
13102   \protect\NoCaseChange
13103   {%
13104     \glsifattribute{#1}{headuc}{true}%
13105     {%
13106       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
13107     }%
13108     {%
13109       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
13110     }%
13111   }%
13112 }
```

sxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
13113 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
13114   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
13115 }
```

Sxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents in all upper case.

```
13116 \newrobustcmd*{\GLSxtrtitlefullpl}[1]{%
13117   \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
13118 }
```

\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.

```
13119 \ifdef\texorpdfstring
13120 {
13121   \newcommand*{\glsfmtshort}[1]{%
13122     \texorpdfstring
13123     {\glsxtrtitleshort{#1}}%
13124     {\glsentryshort{#1}}%
13125   }
13126 }
13127 {
13128   \newcommand*{\glsfmtshort}[1]{%
13129     \glsxtrtitleshort{#1}}
13130 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
13131 \ifdef\texorpdfstring
13132 {
13133   \newcommand*{\glsfmtshortpl}[1]{%
13134     \texorpdfstring
13135       {\glsxtrtitleshortpl{#1}}%
13136       {\glsentryshortpl{#1}}%
13137   }
13138 }
13139 {
13140   \newcommand*{\glsfmtshortpl}[1]{%
13141     \glsxtrtitleshortpl{#1}}
13142 }
```

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).

```
13143 \ifdef\texorpdfstring
13144 {
13145   \newcommand*{\Glsfmtshort}[1]{%
13146     \texorpdfstring
13147       {\Glsxtrtitleshort{#1}}%
13148       {\glsentryshort{#1}}%
13149   }
13150 }
13151 {
13152   \newcommand*{\Glsfmtshort}[1]{%
13153     \Glsxtrtitleshort{#1}}
13154 }
```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```
13155 \ifdef\texorpdfstring
13156 {
13157   \newcommand*{\Glsfmtshortpl}[1]{%
13158     \texorpdfstring
13159       {\Glsxtrtitleshortpl{#1}}%
13160       {\glsentryshortpl{#1}}%
13161   }
13162 }
13163 {
13164   \newcommand*{\Glsfmtshortpl}[1]{%
13165     \Glsxtrtitleshortpl{#1}}
13166 }
```

`\glsfmtname` As above but for the name value.

```
13167 \ifdef\texorpdfstring
```

```

13168 {
13169   \newcommand*{\glsfmtname}[1]{%
13170     \texorpdfstring
13171     {\glsxtrtitlename{#1}}%
13172     {\glsentryname{#1}}%
13173   }
13174 }
13175 {
13176   \newcommand*{\glsfmtname}[1]{%
13177     \glsxtrtitlename{#1}}
13178 }

```

`\Glsfmtname` First letter converted to upper case.

```

13179 \ifdef\texorpdfstring
13180 {
13181   \newcommand*{\Glsfmtname}[1]{%
13182     \texorpdfstring
13183     {\Glsxtrtitlename{#1}}%
13184     {\glsentryname{#1}}%
13185   }
13186 }
13187 {
13188   \newcommand*{\Glsfmtname}[1]{%
13189     \Glsxtrtitlename{#1}}
13190 }

```

`\GLSfmtname` All upper case.

```

13191 \ifdef\texorpdfstring
13192 {
13193   \newcommand*{\GLSfmtname}[1]{%
13194     \texorpdfstring
13195     {\GLSxtrtitlename{#1}}%
13196     {\glsentryname{#1}}%
13197   }
13198 }
13199 {
13200   \newcommand*{\GLSfmtname}[1]{%
13201     \GLSxtrtitlename{#1}}
13202 }

```

`\glsfmttext` As above but for the text value.

```

13203 \ifdef\texorpdfstring
13204 {
13205   \newcommand*{\glsfmttext}[1]{%
13206     \texorpdfstring
13207     {\glsxtrtitletext{#1}}%
13208     {\glsentrytext{#1}}%
13209   }
13210 }

```

```

13211 {
13212   \newcommand*{\glsfmttext}[1]{%
13213     \glsxtrtitletext{#1}}
13214 }

```

`\Glsfmttext` First letter converted to upper case.

```

13215 \ifdef\teorpdfstring
13216 {
13217   \newcommand*{\Glsfmttext}[1]{%
13218     \teorpdfstring
13219     {\Glsxtrtitletext{#1}}%
13220     {\glsentrytext{#1}}%
13221   }
13222 }
13223 {
13224   \newcommand*{\Glsfmttext}[1]{%
13225     \Glsxtrtitletext{#1}}
13226 }

```

`\GLSfmttext` All upper case.

```

13227 \ifdef\teorpdfstring
13228 {
13229   \newcommand*{\GLSfmttext}[1]{%
13230     \teorpdfstring
13231     {\GLSxtrtitletext{#1}}%
13232     {\glsentrytext{#1}}%
13233   }
13234 }
13235 {
13236   \newcommand*{\GLSfmttext}[1]{%
13237     \GLSxtrtitletext{#1}}
13238 }

```

`\glsfmtplural` As above but for the plural value.

```

13239 \ifdef\teorpdfstring
13240 {
13241   \newcommand*{\glsfmtplural}[1]{%
13242     \teorpdfstring
13243     {\glsxtrtitleplural{#1}}%
13244     {\glsentryplural{#1}}%
13245   }
13246 }
13247 {
13248   \newcommand*{\glsfmtplural}[1]{%
13249     \glsxtrtitleplural{#1}}
13250 }

```

`\Glsfmtplural` First letter converted to upper case.

```

13251 \ifdef\teorpdfstring

```

```

13252 {
13253   \newcommand*{\Glsfmtplural}[1]{%
13254     \texorpdfstring
13255     {\Glsxtrtitleplural{#1}}%
13256     {\glsentryplural{#1}}%
13257   }
13258 }
13259 {
13260   \newcommand*{\Glsfmtplural}[1]{%
13261     \Glsxtrtitleplural{#1}}
13262 }

```

`\Glsfmtplural` All upper case.

```

13263 \ifdef\texorpdfstring
13264 {
13265   \newcommand*{\Glsfmtplural}[1]{%
13266     \texorpdfstring
13267     {\Glsxtrtitleplural{#1}}%
13268     {\glsentryplural{#1}}%
13269   }
13270 }
13271 {
13272   \newcommand*{\Glsfmtplural}[1]{%
13273     \Glsxtrtitleplural{#1}}
13274 }

```

`\glsfmtfirst` As above but for the first value.

```

13275 \ifdef\texorpdfstring
13276 {
13277   \newcommand*{\glsfmtfirst}[1]{%
13278     \texorpdfstring
13279     {\glsxtrtitlefirst{#1}}%
13280     {\glsentryfirst{#1}}%
13281   }
13282 }
13283 {
13284   \newcommand*{\glsfmtfirst}[1]{%
13285     \glsxtrtitlefirst{#1}}
13286 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

13287 \ifdef\texorpdfstring
13288 {
13289   \newcommand*{\Glsfmtfirst}[1]{%
13290     \texorpdfstring
13291     {\Glsxtrtitlefirst{#1}}%
13292     {\glsentryfirst{#1}}%
13293   }
13294 }

```

```

13295 {
13296   \newcommand*{\GLSfmtfirst}[1]{%
13297     \Glsxtrtitlefirst{#1}}
13298 }

```

`\GLSfmtfirst` All upper case.

```

13299 \ifdef\teorpdfstring
13300 {
13301   \newcommand*{\GLSfmtfirst}[1]{%
13302     \teorpdfstring
13303       {\GLSxtrtitlefirst{#1}}%
13304       {\glentryfirst{#1}}%
13305   }
13306 }
13307 {
13308   \newcommand*{\GLSfmtfirst}[1]{%
13309     \Glsxtrtitlefirst{#1}}
13310 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

13311 \ifdef\teorpdfstring
13312 {
13313   \newcommand*{\glsfmtfirstpl}[1]{%
13314     \teorpdfstring
13315       {\glxtrtitlefirstplural{#1}}%
13316       {\glentryfirstplural{#1}}%
13317   }
13318 }
13319 {
13320   \newcommand*{\glsfmtfirstpl}[1]{%
13321     \glxtrtitlefirstplural{#1}}
13322 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

13323 \ifdef\teorpdfstring
13324 {
13325   \newcommand*{\Glsfmtfirstpl}[1]{%
13326     \teorpdfstring
13327       {\Glsxtrtitlefirstplural{#1}}%
13328       {\glentryfirstplural{#1}}%
13329   }
13330 }
13331 {
13332   \newcommand*{\Glsfmtfirstpl}[1]{%
13333     \Glsxtrtitlefirstplural{#1}}
13334 }

```

`\GLSfmtfirstpl` All upper case.

```

13335 \ifdef\teorpdfstring

```



```

13336 {
13337   \newcommand*{\GLSfmtfirstpl}[1]{%
13338     \texorpdfstring
13339     {\GLSxtrtitlefirstplural{#1}}%
13340     {\glsentryfirstplural{#1}}%
13341   }
13342 }
13343 {
13344   \newcommand*{\GLSfmtfirstpl}[1]{%
13345     \GLSxtrtitlefirstplural{#1}}
13346 }

```

`\glsfmtlong` As above but for the long value.

```

13347 \ifdef\texorpdfstring
13348 {
13349   \newcommand*{\glsfmtlong}[1]{%
13350     \texorpdfstring
13351     {\glsxtrtitlelong{#1}}%
13352     {\glsentrylong{#1}}%
13353   }
13354 }
13355 {
13356   \newcommand*{\glsfmtlong}[1]{%
13357     \glsxtrtitlelong{#1}}
13358 }

```

`\Glsfmtlong` First letter converted to upper case.

```

13359 \ifdef\texorpdfstring
13360 {
13361   \newcommand*{\Glsfmtlong}[1]{%
13362     \texorpdfstring
13363     {\Glsxtrtitlelong{#1}}%
13364     {\glsentrylong{#1}}%
13365   }
13366 }
13367 {
13368   \newcommand*{\Glsfmtlong}[1]{%
13369     \Glsxtrtitlelong{#1}}
13370 }

```

`\GLSfmtlong` All upper case.

```

13371 \ifdef\texorpdfstring
13372 {
13373   \newcommand*{\GLSfmtlong}[1]{%
13374     \texorpdfstring
13375     {\GLSxtrtitlelong{#1}}%
13376     {\glsentrylong{#1}}%
13377   }
13378 }

```

```

13379 {
13380   \newcommand*{\GLSfmtlong}[1]{%
13381     \GLSxtrtitlelong{#1}}
13382 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

13383 \ifdef\texorpdfstring
13384 {
13385   \newcommand*{\glsfmtlongpl}[1]{%
13386     \texorpdfstring
13387       {\glsxtrtitlelongpl{#1}}%
13388       {\glsentrylongpl{#1}}%
13389   }
13390 }
13391 {
13392   \newcommand*{\glsfmtlongpl}[1]{%
13393     \glsxtrtitlelongpl{#1}}
13394 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

13395 \ifdef\texorpdfstring
13396 {
13397   \newcommand*{\Glsfmtlongpl}[1]{%
13398     \texorpdfstring
13399       {\Glsxtrtitlelongpl{#1}}%
13400       {\glsentrylongpl{#1}}%
13401   }
13402 }
13403 {
13404   \newcommand*{\Glsfmtlongpl}[1]{%
13405     \Glsxtrtitlelongpl{#1}}
13406 }

```

`\GLSfmtlongpl` All upper case.

```

13407 \ifdef\texorpdfstring
13408 {
13409   \newcommand*{\GLSfmtlongpl}[1]{%
13410     \texorpdfstring
13411       {\GLSxtrtitlelongpl{#1}}%
13412       {\glsentrylongpl{#1}}%
13413   }
13414 }
13415 {
13416   \newcommand*{\GLSfmtlongpl}[1]{%
13417     \GLSxtrtitlelongpl{#1}}
13418 }

```

`\glspdffmtfull` Can't use `\glsxtrinlinefullformat` in PDF bookmarks as it's not fully expandable. This command is for the PDF part of `\texorpdfstring` for the full form.

```

13419 \newcommand*{\glspdffmtfull}[1]{\glsentrylong{#1} (\glsentryshort{#1})}%
glspdffmtfullpl Likewise for plural.
13420 \newcommand*{\glspdffmtfullpl}[1]{\glsentrylongpl{#1} (\glsentryshortpl{#1})}%

\glfmtfull In-line full format.
13421 \ifdef\texorpdfstring
13422 {
13423   \newcommand*{\glfmtfull}[1]{%
13424     \texorpdfstring
13425     {\glxtrtitlefull{#1}}%
13426     {\glspdffmtfull{#1}}%
13427   }
13428 }
13429 {
13430   \newcommand*{\glfmtfull}[1]{%
13431     \glxtrtitlefull{#1}}
13432 }

\Glsfmtfull First letter converted to upper case.
13433 \ifdef\texorpdfstring
13434 {
13435   \newcommand*{\Glsfmtfull}[1]{%
13436     \texorpdfstring
13437     {\Glsxtrtitlefull{#1}}%
13438     {\glspdffmtfull{#1}{}}%
13439   }
13440 }
13441 {
13442   \newcommand*{\Glsfmtfull}[1]{%
13443     \Glsxtrtitlefull{#1}}
13444 }

\GLSfmtfull All upper case.
13445 \ifdef\texorpdfstring
13446 {
13447   \newcommand*{\GLSfmtfull}[1]{%
13448     \texorpdfstring
13449     {\GLSxtrtitlefull{#1}}%
13450     {\glspdffmtfull{#1}}%
13451   }
13452 }
13453 {
13454   \newcommand*{\GLSfmtfull}[1]{%
13455     \GLSxtrtitlefull{#1}}
13456 }

\glfmtfullpl In-line full plural format.

```

```

13457 \ifdef\teorpdfstring
13458 {
13459   \newcommand*\glsfmtfullpl}[1]{%
13460     \teorpdfstring
13461     {\glsxtrtitlefullpl{#1}}%
13462     {\glspdfdfmtfullpl{#1}}%
13463   }
13464 }
13465 {
13466   \newcommand*\glsfmtfullpl}[1]{%
13467     \glsxtrtitlefullpl{#1}}
13468 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

13469 \ifdef\teorpdfstring
13470 {
13471   \newcommand*\Glsfmtfullpl}[1]{%
13472     \teorpdfstring
13473     {\Glsxtrtitlefullpl{#1}}%
13474     {\glspdfdfmtfullpl{#1}{}}%
13475   }
13476 }
13477 {
13478   \newcommand*\Glsfmtfullpl}[1]{%
13479     \Glsxtrtitlefullpl{#1}}
13480 }

```

`\GLSfmtfullpl` All upper case.

```

13481 \ifdef\teorpdfstring
13482 {
13483   \newcommand*\GLSfmtfullpl}[1]{%
13484     \teorpdfstring
13485     {\GLSxtrtitlefullpl{#1}}%
13486     {\glspdfdfmtfullpl{#1}{}}%
13487   }
13488 }
13489 {
13490   \newcommand*\GLSfmtfullpl}[1]{%
13491     \GLSxtrtitlefullpl{#1}}
13492 }

```

1.9 Multi (Combined/Compound) Entries

(I'd rather call these combined or compound entries but `\cgl` is already taken.)

New to version 1.48, the commands here provide a way of referencing multiple entries as a single unit. For example, biological organisms are often referred to by their genus and species, such as *Clostridium botulinum* and *Clostridium perfringens* (where the genus is *Clostridium*).

The genus is often abbreviated after first use, regardless of which species in the genus is being referenced. For example, “*Clostridium botulinum* and *C. perfringens*”. This can’t be supported by any abbreviation styles unless the genus and species names are defined separately. For example:

```
\setabbreviationstyle{long-only-short-only}
\newabbreviation{clostridium}{C.}{Clostridium}
\newglossaryentry{botulinum}{name={botulinum},description={}}
\newglossaryentry{perfringens}{name={perfringens},description={}}
```

This means that the entries then need to be referenced using a rather cumbersome method:

```
\gls{clostridium} \gls{botulinum} and \gls{clostridium}
\gls{perfringens}
```

This section provides a command that will provide a way of defining a label that represents a combination of entries (which must all be first defined). For example:

```
\multiglossaryentry{cbot}{clostridium,botulinum}
```

This label can then be referenced using `\mgls`, which internally uses `\gls` for each component. The last component in the list is considered the “main” component (not to be confused with the main glossary). If this isn’t the case, the label of the main component should be added in the optional argument before the label list. Note that the multi-label (cbot in this case) can’t be referenced using commands like `\gls`.

First define the general set of options that should be applied to all multi-entries. These can be set with:

ssaryentrysetup

```
13493 \newcommand*{\multiglossaryentrysetup}[1]{\setkeys{glxstrcombined}{#1}}
```

combined@indexmain Numeric value: 0=false (don’t index main component), 1=true (always index main component), 2=first (only index main component on first use). Default: 1 (true);

```
13494 \newcommand*{\@gls@combined@indexmain}{1}
13495 \define@choicekey{glxstrcombined}{indexmain}%
13496 [\@gls@combined@indexmain@val\@gls@combined@indexmain]
13497 {false,true,first}[true]{}
```

ned@indexothers Numeric value: 0=false (don’t index other components), 1=true (always index other components), 2=first (only index other components on first use). Default: 2 (first);

```
13498 \newcommand*{\@gls@combined@indexothers}{2}
13499 \define@choicekey{glxstrcombined}{indexothers}%
13500 [\@gls@combined@indexothers@val\@gls@combined@indexothers]
13501 {false,true,first}[true]{}
```

@combined@hyper Numeric value: 0=none (`\mgls` doesn’t create a hyperlink), 1=allmain (all content hyperlinks to the main component), 2=mainonly (only the main component has a hyperlink), 3=individual (each component has a hyperlink to their own target). Default: 3.

```

13502 \newcommand*{\@gls@combined@hyper}{3}
13503 \define@choicekey{glsxtrcombined}{hyper}{%
13504   [\@gls@combined@hyper@val\@gls@combined@hyper]
13505   {none,allmain,mainonly,individual,otheronly,notmainfirst,nototherfirst,notfirst}{}}

combined@encapmain  Location encap value for main component (corresponding to format key in \gls).
13506 \newcommand*{\@gls@combined@encapmain}{glsnumberformat}
13507 \define@key{glsxtrcombined}{encapmain}{%
13508   \renewcommand*{\@gls@combined@encapmain}{#1}%
13509 }

combined@encapothers  Location encap value for other components (corresponding to format key in \gls).
13510 \newcommand*{\@gls@combined@encapothers}{glsnumberformat}
13511 \define@key{glsxtrcombined}{encapothers}{%
13512   \renewcommand*{\@gls@combined@encapothers}{#1}%
13513 }

combined@textformat  Encapsulate entire content with the command identified by the given control sequence
name.
13514 \newcommand*{\@gls@combined@textformat}{@firstofone}
13515 \define@key{glsxtrcombined}{textformat}{%
13516   \renewcommand*{\@gls@combined@textformat}{#1}%
13517 }

combined@category  Assign a category to the combined set.
13518 \newcommand*{\@gls@combined@category}{}
13519 \define@key{glsxtrcombined}{category}{%
13520   \renewcommand*{\@gls@combined@category}{#1}%
13521 }

    Pre-options family:
13522 \define@key{glsxtrcombinedpreopts}{category}{%
13523   \renewcommand*{\@gls@combined@category}{#1}%
13524 }

combined@mglsopts  Default options to pass to \mgl.
13525 \newcommand*{\@gls@combined@mglsopts}{}
13526 \define@key{glsxtrcombined}{mglsopts}{%
13527   \renewcommand*{\@gls@combined@mglsopts}{#1}%
13528 }
13529 \define@key{glsxtrcombinedpreopts}{mglsopts}{%
13530   \@gls@combined@mglsopts@do
13531   {%
13532     \renewcommand*{\@gls@combined@mglsopts}{#1}%
13533   }%
13534 }

combined@mglsopts@do
13535 \newcommand*{\@gls@combined@mglsopts@do}[1]{#1}

```

isable@mglsopts

```
13536 \newcommand*{\mgl@disable@mglsopts}{%
13537 \let\@gls@combined@mglsopts@do\@gls@combined@mglsopts@do@not
13538 }
```

enable@mglsopts

```
13539 \newcommand*{\mgl@enable@mglsopts}{%
13540 \let\@gls@combined@mglsopts@do\@firstofone
13541 }
```

ned@mglsopts@do

```
13542 \newcommand*{\@gls@combined@mglsopts@do@not}[1]{%
13543 \PackageError{glossaries-extra}{‘mglsopts’ key not permitted inside
13544 ‘setup’ value}{}%
13545 }
```

ned@firstprefix Prefix for multi-entry first use.

```
13546 \newcommand*{\@gls@combined@firstprefix}{%
13547 \define@key{glsxtrcombined}{firstprefix}{%
13548 \renewcommand*{\@gls@combined@firstprefix}{#1}%
13549 }
```

ined@usedprefix Prefix for multi-entry subsequent first use.

```
13550 \newcommand*{\@gls@combined@usedprefix}{%
13551 \define@key{glsxtrcombined}{usedprefix}{%
13552 \renewcommand*{\@gls@combined@usedprefix}{#1}%
13553 }
```

ned@firstsuffix Suffix for multi-entry first use.

```
13554 \newcommand*{\@gls@combined@firstsuffix}{%
13555 \define@key{glsxtrcombined}{firstsuffix}{%
13556 \renewcommand*{\@gls@combined@firstsuffix}{#1}%
13557 }
```

ined@usedsuffix Suffix for multi-entry subsequent first use.

```
13558 \newcommand*{\@gls@combined@usedsuffix}{%
13559 \define@key{glsxtrcombined}{usedsuffix}{%
13560 \renewcommand*{\@gls@combined@usedsuffix}{#1}%
13561 }
```

d@firstskipmain Skip the main element on first use (multi-entry first use not element first use).

```
13562 \define@boolkey{glsxtrcombined}{firstskipmain}[true]{%
13563 \KV@glsxtrcombined@firstskipmainfalse
```

firstskipothers Skip the other elements on first use (multi-entry first use not element first use).

```
13564 \define@boolkey{glsxtrcombined}{firstskipothers}[true]{%
13565 \KV@glsxtrcombined@firstskipothersfalse
```

ed@usedskipmain Skip the main element on subsequent use (multi-entry subsequent use not element subsequent use).

```

13566 \define@boolkey{glxstrcombined}{usedskipmain}[true]{}
13567 \KV@glxstrcombined@usedskipmainfalse

```

@usedskipothers Skip the other elements on subsequent use (multi-entry subsequent use not element subsequent use).

```

13568 \define@boolkey{glxstrcombined}{usedskipothers}[true]{}
13569 \KV@glxstrcombined@usedskipothersfalse

```

combined@postlinks Determine whether or not to use the individual element post-link hooks.

```

13570 \newcommand*{\@gls@combined@postlinks@nr}{0}
13571 \define@choicekey{glxstrcombined}{postlinks}%
13572 [\@gls@combined@postlinks@val\@gls@combined@postlinks@nr]
13573 {none,all,notlast,mainnotlast,mainonly,othernotlast,otheronly}{}

```

combined@mpostlink Determine whether or not to use the multi-entry post-link hook.

```

13574 \newcommand*{\@gls@combined@mpostlink@nr}{1}
13575 \define@choicekey{glxstrcombined}{mpostlink}%
13576 [\@gls@combined@mpostlink@val\@gls@combined@mpostlink@nr]
13577 {false,true,firstonly,usedonly}[true]{}

```

postlinkelement Determine which element to use for the post-link hook.

```

13578 \newcommand*{\@gls@combined@mpostlinkelement@nr}{0}
13579 \define@choicekey{glxstrcombined}{mpostlinkelement}%
13580 [\@gls@combined@mpostlinkelement@val\@gls@combined@mpostlinkelement@nr]
13581 {last,main,custom}{}

```

\glxstrifmulti

```

13582 \newcommand*{\glxstrifmulti}[3]{\ifcsdef{@gls@combined@#1@main}{#2}{#3}}

```

glxstrmultimain

```

13583 \newcommand*{\glxstrmultimain}[1]{\csuse{@gls@combined@#1@main}}

```

glxstrmultilist

```

13584 \newcommand*{\glxstrmultilist}[1]{\csuse{@gls@combined@#1@list}}

```

totalelements Total number of elements.

```

13585 \newcommand*{\glxstrmultitotalelements}[1]{\csuse{@gls@combined@#1@total}}

```

multimainindex Index of main element (starting from 1). If the main element is the last element in the list then this should equal the total number of elements.

```

13586 \newcommand*{\glxstrmultimainindex}[1]{\csuse{@gls@combined@#1@mainindex}}

```

lastotherindex Index of the last non-main element.

```

13587 \newcommand*{\glxstrmultilastotherindex}[1]{\csuse{@gls@combined@#1@lastotherindex}}

```


saryentryglobal Make definitions global.

```
13588 \newif\ifmultiglossaryentryglobal
13589 \multiglossaryentryglobalfalse
```

glselementindex Count register to keep track of the current element index.

```
13590 \newcount\mglselementindex
```

tiglossaryentry

`\multiglossaryentry[<options>]{<multi-label>}[<main label>]{<label list>}`

Defines the label *<multi-label>* that can be used in `\mgls`.

```
13591 \newrobustcmd{\multiglossaryentry}[1][{}]{%
13592   \def\@gls@combined@current@opts{#1}%
13593   \ifnum\@glsxtr@docdefval=1\relax
13594     \let\@multi@glossentry@donext\@defmultiglossaryentry
13595   \else
13596     \let\@multi@glossentry@donext\@multiglossaryentry
13597   \fi
13598   \@multi@glossentry@donext
13599 }
```

tiglossaryentry

```
13600 \newcommand*{\@multiglossaryentry}[1]{%
13601   \def\@gls@combined@current@label{#1}%
13602   \@multi@glossaryentry
13603 }
```

i@glossaryentry Check for existence.

```
13604 \newcommand*{\@multi@glossaryentry}[2][{}]{%
13605   \ifcsdef{\@gls@combined@\@gls@combined@current@label @main}%
13606     {\PackageError{glossaries-extra}%
13607       {Multi-entry label ‘\@gls@combined@current@label’ already defined}%
13608       {}%
13609     }%
13610   {%
13611     \@multi@glossary@entry{#1}{#2}%
13612   }%
13613 }
```

tiglossaryentry Used if document definitions are on.

```
13614 \newcommand*{\@defmultiglossaryentry}[1]{%
13615   \def\@gls@combined@current@label{#1}%
13616   \@def@multi@glossaryentry
13617 }
```

i@glossaryentry Used if document definitions are on.

```

13618 \newcommand*{\@def@multi@glossaryentry}[2] [] {%
13619   \let\@def@multi@glossaryentry@do\@multi@glossary@entry
13620   \ifundef\@glstr@docdefs@multilist
13621   {%
13622     \gdef\@glstr@docdefs@multilist{}%
13623     \listxadd
13624       {\@glstr@docdefs@multilist}{\expandonce\@gl@combined@current@label}%
13625   }%
13626   {%
13627     \xifinlist{\@gl@combined@current@label}{\@glstr@docdefs@multilist}%
13628     {%
13629       \PackageError{glossaries-extra}%
13630         {Multi-entry label ‘\@gl@combined@current@label’ already defined}%
13631         {}%
13632       \let\@def@multi@glossaryentry@do\@gobbletwo
13633     }%
13634     {%
13635       \listxadd
13636         {\@glstr@docdefs@multilist}{\expandonce\@gl@combined@current@label}%
13637     }%
13638   }%
13639   \@def@multi@glossaryentry@do{#1}{#2}%
13640 }

```

sary@doifexists

```

13641 \newcommand*{\@multi@glossary@doifexists}{\gl@doifexists}

```

ti@glossaryentry

```

\providemultiglossaryentry[<options>]{<multi-label>}{<main label>}{<label
list>}

```

Defines a multi-entry unless it has already been defined.

```

13642 \newrobustcmd{\providemultiglossaryentry}[2] [] {%
13643   \def\@gl@combined@current@opts{#1}%
13644   \def\@gl@combined@current@label{#2}%
13645   \ifcsdef\@gl@combined@\@gl@combined@current@label @main}%
13646   {\def\@multi@glossentry@donext{\@provide@multi@glossaryentry@noop}}%
13647   {%
13648     \ifnum\@glstr@docdefval=1\relax
13649     \def\@multi@glossentry@donext{\@def@multi@glossaryentry}%
13650   \else
13651     \def\@multi@glossentry@donext{\@multi@glossaryentry}%
13652   \fi
13653 }%
13654 \@multi@glossentry@donext
13655 }

```

ssaryentry@noop Do nothing.

```
13656 \newcommand*{\@provide@multi@glossaryentry@noop}[2][{}]
```

ssaryentry@list List of all defined multi-entry sets.

```
13657 \newcommand*{\@multi@glossaryentry@list}{}%
```

@glossary@entry

```
13658 \newcommand*{\@multi@glossary@entry}[2]{%
```

```
13659 \protected@edef\@gls@combined@current@main{#1}%
```

Fully expand list.

```
13660 \protected@edef\@gls@combined@currentlist{#2}%
```

Count items in list, check they are all defined, and find last item at the same time.

```
13661 \mglselementindex=0\relax
```

```
13662 \@for\@gls@tmp:=\@gls@combined@currentlist\do{%
```

```
13663 \advance\mglselementindex by 1\relax
```

```
13664 \@multi@glossary@doifexists{\@gls@tmp}{}%
```

```
13665 \let\@gls@combined@finalitem\@gls@tmp
```

```
13666 \ifdefvoid\@gls@combined@current@main
```

```
13667 {}%
```

```
13668 {%
```

```
13669 \ifx\@gls@combined@current@main\@gls@tmp
```

```
13670 \ifmultiglossaryentryglobal
```

```
13671 \global\cslet{\@gls@combined@\@gls@combined@current@label @main}%
```

```
13672 \@gls@combined@current@main
```

```
13673 \csxdef{\@gls@combined@\@gls@combined@current@label @mainindex}%
```

```
13674 {\the\mglselementindex}%
```

```
13675 \else
```

```
13676 \cslet{\@gls@combined@\@gls@combined@current@label @main}%
```

```
13677 \@gls@combined@current@main
```

```
13678 \csedef{\@gls@combined@\@gls@combined@current@label @mainindex}%
```

```
13679 {\the\mglselementindex}%
```

```
13680 \fi
```

```
13681 \else
```

```
13682 \ifmultiglossaryentryglobal
```

```
13683 \csxdef{\@gls@combined@\@gls@combined@current@label @lastotherindex}%
```

```
13684 {\the\mglselementindex}%
```

```
13685 \else
```

```
13686 \csedef{\@gls@combined@\@gls@combined@current@label @lastotherindex}%
```

```
13687 {\the\mglselementindex}%
```

```
13688 \fi
```

```
13689 \fi
```

```
13690 }%
```

```
13691 }%
```

```
13692 \ifmultiglossaryentryglobal
```

```
13693 \csxdef{\@gls@combined@\@gls@combined@current@label @total}%
```

```
13694 {\the\mglselementindex}%
```

```
13695 \else
```

```
13696 \csedef{\@gls@combined@\@gls@combined@current@label @total}%
```

```

13697         {\the\mglselementindex}%
13698     \fi
13699     \ifnum\mglselementindex<2\relax
13700         \PackageError{glossaries-extra}{At least 2 labels required in
13701             multi-entry element list (\number\mglselementindex\space found)}{ }%
13702     \else
13703         \ifdefvoid\@gls@combined@current@main
13704             {}%
13705             {%
                If \@gls@combined@(\label)@main hasn't been set then it wasn't included in the list.
13706             \ifcsundef{\@gls@combined@\@gls@combined@current@label @main}%
13707             {\PackageError{glossaries-extra}%
13708                 {Main element '@@gls@combined@current@main' not found in list}%
13709                 {The final element '@@gls@combined@finalitem' will be used instead}}
                Set to empty so that the default (final element) is used instead.
13710             \let\@gls@combined@current@main\@empty
13711             }%
13712             {}%
13713             }%
13714             \ifdefvoid\@gls@combined@current@main
13715                 {%
                    Set main to final element.
13716                 \ifmultiglossaryentryglobal
13717                     \global\cslet{\@gls@combined@\@gls@combined@current@label @main}%
13718                     \@gls@combined@finalitem
13719                     \global\csletcs{\@gls@combined@\@gls@combined@current@label @mainindex}%
13720                     {\@gls@combined@\@gls@combined@current@label @total}%
13721                     \csxdef{\@gls@combined@\@gls@combined@current@label @lastotherindex}%
13722                     {\the\numexpr\mglselementindex-1 }%
13723                 \else
13724                     \cslet{\@gls@combined@\@gls@combined@current@label @main}%
13725                     \@gls@combined@finalitem
13726                     \csletcs{\@gls@combined@\@gls@combined@current@label @mainindex}%
13727                     {\@gls@combined@\@gls@combined@current@label @total}%
13728                     \csedef{\@gls@combined@\@gls@combined@current@label @lastotherindex}%
13729                     {\the\numexpr\mglselementindex-1 }%
13730                 \fi
13731             }%
13732             {}%
13733             \ifmultiglossaryentryglobal
                Globally define element list.
13734             \global\cslet{\@gls@combined@\@gls@combined@current@label @list}%
13735             \@gls@combined@currentlist
                Globally define options.
13736             \protected\csxdef{\@gls@combined@\@gls@combined@current@label @options}%
13737             {\@gls@combined@current@opts}%

```

Global conditional definition.

```

13738 \expandafter\ifdefinable
13739 \csname if@glsc@combined@\@glsc@combined@current@label @flag\endcsname
13740 {\expandafter\global\expandafter
13741 \newif\csname if@glsc@combined@\@glsc@combined@current@label @flag\endcsname}%
13742 \expandafter\global
13743 \csname @glsc@combined@\@glsc@combined@current@label @flagfalse\endcsname
13744 \else

```

Locally define element list.

```

13745 \cslet{@glsc@combined@\@glsc@combined@current@label @list}%
13746 \@glsc@combined@currentlist

```

Locally define options.

```

13747 \protected@csedef{@glsc@combined@\@glsc@combined@current@label @options}%
13748 {\@glsc@combined@current@opts}%

```

Local conditional definition.

```

13749 \newboolean{@glsc@combined@\@glsc@combined@current@label @flag}%
13750 \csname @glsc@combined@\@glsc@combined@current@label @flagfalse\endcsname
13751 \fi
13752 \fi
13753 \writemultiglossentry
13754 {\@glsc@combined@current@opts}{\@glsc@combined@current@label}%
13755 {\csuse{@glsc@combined@\@glsc@combined@current@label @main}}{#2}%

```

Append label to list.

```

13756 \ifmultiglossaryentryglobal
13757 \ifdefempty\@multi@glossaryentry@list
13758 {\let\@multi@glossaryentry@list\@glsc@combined@current@label}%
13759 {%
13760 \eappto\@multi@glossaryentry@list{,\expandonce\@glsc@combined@current@label}%
13761 }%
13762 \else
13763 \ifdefempty\@multi@glossaryentry@list
13764 {\global\let\@multi@glossaryentry@list\@glsc@combined@current@label}%
13765 {%
13766 \xappto\@multi@glossaryentry@list{,\expandonce\@glsc@combined@current@label}%
13767 }%
13768 \fi
13769 }

```

sxtr@multientry

`\@glssxtr@multientry{<options>}{<multilabel>}{<main>}{<list>}`

Information for aux file. Useful for bib2gls and also for docdef.

```

13770 \newcommand*{\@glssxtr@multientry}[4]{%
13771 \ifnum\@glssxtr@docdefval=1\relax

```

```

13772 \bgroup
13773 \def\@gls@combined@current@opts{#1}%
13774 \def\@gls@combined@current@label{#2}%
13775 \let\@multi@glossary@doifexists\@secondoftwo
13776 \let\writemultiglossentry\@gobblefour
13777 \multiglossaryentryglobaltrue
13778 \@multi@glossary@entry{#3}{#4}%
13779 \egroup
13780 \fi
13781 }

```

`\multiglossentry` This can be redefined to do nothing if the information isn't required.

```

13782 \newcommand*\writemultiglossentry[4]{%
13783 \protected@write\@auxout{}\string\@glsxtr@multientry{#1}{#2}{#3}{#4}}%
13784 }

```

`\ifmglsused` Determines whether or not the multi-entry set has been referenced by commands like `\mgls` or `\mglsname`.

```

13785 \newcommand*\ifmglsused[3]{%
13786 \ifbool{\@gls@combined@#1@flag}{#2}{#3}%
13787 }

```

`\mglsunset` Unset the flag.

```

13788 \newcommand*\mglsunset[1]{%
13789 \gls@ifnotmeasuring
13790 {%
13791 \glsxtrifmulti{#1}{\@mglsunset{#1}}%
13792 {%
13793 \glsxtrundefaction{Multi entry ‘#1’ hasn’t been defined}%
13794 {You need to define ‘#1’ with \string\multiglossaryentry}%
13795 }%
13796 }%
13797 }

```

`\@mglsunset`

```

13798 \newcommand*\@mglsunset[1]{%
13799 \expandafter\global\csname @gls@combined@#1@flagtrue\endcsname
13800 }

```

`\mglsreset` Unset the flag.

```

13801 \newcommand*\mglsreset[1]{%
13802 \gls@ifnotmeasuring
13803 {%
13804 \glsxtrifmulti{#1}{\@mglsreset{#1}}%
13805 {%
13806 \glsxtrundefaction{Multi entry ‘#1’ hasn’t been defined}%
13807 {You need to define ‘#1’ with \string\multiglossaryentry}%
13808 }%

```

```

13809 }%
13810 }

\@mglsreset
13811 \newcommand*{\@mglsreset}[1]{%
13812 \expandafter\global\csname @gls@combined@#1@flagfalse\endcsname
13813 }

\mglslocalunset  Unset the flag.
13814 \newcommand*{\mglslocalunset}[1]{%
13815 \gls@ifnotmeasuring
13816 {%
13817 \glstrifmulti{#1}{\@mglslocalunset{#1}}%
13818 {%
13819 \glstrundefaction{Multi entry ‘#1’ hasn’t been defined}%
13820 {You need to define ‘#1’ with \string\multiglossaryentry}%
13821 }%
13822 }%
13823 }

@mglsllocalunset
13824 \newcommand*{\@mglsllocalunset}[1]{%
13825 \csname @gls@combined@#1@flagtrue\endcsname
13826 }

\mglslocalreset  Unset the flag.
13827 \newcommand*{\mglslocalreset}[1]{%
13828 \gls@ifnotmeasuring
13829 {%
13830 \glstrifmulti{#1}{\@mglsllocalreset{#1}}%
13831 {%
13832 \glstrundefaction{Multi entry ‘#1’ hasn’t been defined}%
13833 {You need to define ‘#1’ with \string\multiglossaryentry}%
13834 }%
13835 }%
13836 }

@mglsllocalreset
13837 \newcommand*{\@mglsllocalreset}[1]{%
13838 \csname @gls@combined@#1@flagfalse\endcsname
13839 }

\mglsunsetall  Unset all.
13840 \newcommand*{\mglsunsetall}{%
13841 \@for\@mgls@thislabel:=\@multi@glossaryentry@list\do{\mglsunset\@mgls@thislabel}%
13842 }%

```

\mglsresetall Reset all.

```
13843 \newcommand*{\mglsresetall}{%
13844   \for\@mgls@thislabel:=\@multi@glossaryentry@list\do{\mglsreset\@mgls@thislabel}%
13845 }%
```

\mglsSetMain

\mglsSetName{<multi-label>}{<new main>}

Allow the main label to be changed (local).

```
13846 \newrobustcmd{\mglsSetMain}[2]{%
13847   \ifcsundef{@gls@combined@#1@main}%
13848   {\PackageError{glossaries-extra}{Multi-entry label ‘#1’ not defined}{}}%
13849   {%
13850     \protected@edef\@gls@combined@current@main{#2}%
13851     \letcs\@gls@combined@currentlist{\@gls@combined@#1@list}%

```

Check that the given label is in the list of elements and update main and last other element index.

```
13852   \mglselementindex=0\relax
13853   \count@=0\relax
13854   \@for\@gls@tmp:=\@gls@combined@currentlist\do{%
13855     \advance\mglselementindex by 1\relax
13856     \ifx\@gls@combined@current@main\@gls@tmp
13857       \count@=\mglselementindex\relax
13858       \let\@gls@combined@finalitem\@gls@tmp
13859       \ifmultiglossaryentryglobal
13860         \global\cslet{\@gls@combined@#1@main}\@gls@combined@current@main
13861         \csxdef{\@gls@combined@#1@mainindex}{\the\mglselementindex}%
13862       \else
13863         \cslet{\@gls@combined@#1@main}\@gls@combined@current@main
13864         \csedef{\@gls@combined@#1@mainindex}{\the\mglselementindex}%
13865       \fi
13866     \else
13867       \ifmultiglossaryentryglobal
13868         \csxdef{\@gls@combined@#1@lastotherindex}{\the\mglselementindex}%
13869       \else
13870         \csedef{\@gls@combined@#1@lastotherindex}{\the\mglselementindex}%
13871       \fi
13872     \fi
13873   }%
13874   \ifnum\count@=0\relax
13875     \PackageError{glossaries-extra}{Label ‘#2’ is not in ‘#1’ set
13876       (\@gls@combined@currentlist)}{}%

```

Default to final item.

```
13877   \ifmultiglossaryentryglobal
13878     \global\cslet{\@gls@combined@#1@main}\@gls@combined@finalitem

```



```

13879      \csxdef{@gls@combined@#1@mainindex}{\the\mglselementindex}%
13880      \csxdef{@gls@combined@#1@lastotherindex}{%
13881        \number\numexpr\mglselementindex-1 }%
13882      \else
13883        \cslet{@gls@combined@#1@main}\@gls@combined@finalitem
13884        \csedef{@gls@combined@#1@mainindex}{\the\mglselementindex}%
13885        \csedef{@gls@combined@#1@lastotherindex}{%
13886          \number\numexpr\mglselementindex-1 }%
13887      \fi
13888    \fi
13889  }%
13890 }

```

\mglSetOptions

`\mglSetOptions{<multi-label>}{<new options>}`

Allow the options to be changed (local). No expansion is applied.

```

13891 \newrobustcmd{\mglSetOptions}[2]{%
13892   \ifcsundef{@gls@combined@#1@main}%
13893   {\PackageError{glossaries-extra}{Multi-entry label ‘#1’ not defined}{}}%
13894   {%
13895     \csdef{@gls@combined@#1@options}{#2}%
13896   }%
13897 }

```

\mglAddOptions

`\mglAddOptions{<multi-label>}{<extra options>}`

Allow the options to be changed (local). No expansion is applied.

```

13898 \newrobustcmd{\mglAddOptions}[2]{%
13899   \ifcsundef{@gls@combined@#1@main}%
13900   {\PackageError{glossaries-extra}{Multi-entry label ‘#1’ not defined}{}}%
13901   {%
13902     \ifcsemtyp{@gls@combined@#1@options}%
13903     {\csdef{@gls@combined@#1@options}{#2}}%
13904     {\csappto{@gls@combined@#1@options}{, #2}}%
13905   }%
13906 }

```

Options for \mgl:

\@mgl@all Options to apply to all elements.

```

13907 \newcommand*{\@mgl@all}{}
13908 \define@key{mgl}{all}{\renewcommand*{\@mgl@all}{#1}}

```

```

\@mglmain Options to apply to the main element only.
13909 \newcommand*{\@mglmain}{%
13910 \define@key{mgl}{main}{\renewcommand*{\@mglmain}{#1}}

\@mgl{others} Options to apply to the other (no main) elements.
13911 \newcommand*{\@mgl{others}}{%
13912 \define@key{mgl}{others}{\renewcommand*{\@mgl{others}}{#1}}

\@mgl{setup} Options to apply to \multiglossaryentrysetup.
13913 \newcommand*{\@mgl{setup}}{%
13914 \define@key{mgl}{setup}{%
13915 \@mgl{setup}@do{\renewcommand*{\@mgl{setup}}{#1}}%
13916 }

\@mgl{setup}@do
13917 \newcommand*{\@mgl{setup}@do}[1]{#1}

ls@setup@do@not
13918 \newcommand*{\@mgl{setup}@do@not}[1]{%
13919 \PackageError{glossaries-extra}{‘setup’ key not permitted inside
13920 ‘mglsopts’ value}{}%
13921 }

s@disable@setup
13922 \newcommand*{\mgl{disable@setup}}{%
13923 \let\@mgl{setup}@do\@mgl{setup}@do@not
13924 }

ls@enable@setup
13925 \newcommand*{\mgl{enable@setup}}{%
13926 \let\@mgl{setup}@do\@firstofone
13927 }

gls@unsetaction
13928 \newcommand\@mgl{unsetaction}{0}
13929 \define@choicekey{mgl}{multiunset}[\@mgl{unsetaction}@val\@mgl{unsetaction}]%
13930 {global,local,none}{}

gls@presetlocal
13931 \define@boolkey{mgl}{presetlocal}[true]{}
13932 \KV@mgl{presetlocal}false

\@mgl{hyper}
13933 \newcommand*{\@mgl{hyper}}{%
13934 \define@choicekey{mgl}{hyper}[\@mgl{hyper}@val\@mgl{hyper}@nr]{true,false}[true]%
13935 {%
13936 \renewcommand*{\@mgl{hyper}}{hyper=#1}%
13937 \ifnum\@mgl{hyper}@nr=1\relax

```

```

13938 \let\@mgls@hyperlink\@secondoftwo
13939 \else
13940 \let\@mgls@hyperlink\@mgls@hyperlink
13941 \fi
13942 }

```

@mgls@hyperlink

```

13943 \newcommand*\@mgls@hyperlink}[2]{%
13944 \ifx\@glslink\glsdonohyperlink
13945 #2%
13946 \else
13947 \glsxtr@org@dohyperlink{\gloslinkprefix#1}{#2}%
13948 \fi
13949 }

```

@mgls@hyperlink

```

13950 \let\@mgls@hyperlink\@mgls@hyperlink

```

mglsforelements

```
\mglsforelements{\multi-label}{\cs}{\body}
```

```

13951 \newcommand*\mglsforelements}[3]{%
13952 \expandafter\@for\expandafter#2\expandafter:\expandafter
13953 =\csname @gls@combined@#1@list\endcsname\do{#3}%
13954 }

```

orotherelements

```
\mglsforotherelements{\multi-label}{\cs}{\body}
```

```

13955 \newcommand*\mglsforotherelements}[3]{%
13956 \expandafter\@for\expandafter#2\expandafter:\expandafter
13957 =\csname @gls@combined@#1@list\endcsname\do
13958 {\expandafter\ifdefequal\csname @gls@combined@#1@main\endcsname{#2}{\}{#3}}%
13959 }

```

mglsunsetothers

```

13960 \newcommand*\mglsunsetothers}[1]{%
13961 \mglsforotherelements{#1}{\@gls@tmp}{\glsunset{\@gls@tmp}}%
13962 }

```

ocalunsetothers

```

13963 \newcommand*\mglslocalunsetothers}[1]{%
13964 \mglsforotherelements{#1}{\@gls@tmp}{\glslocalunset{\@gls@tmp}}%
13965 }

```

glselementreset

```
13966 \newcommand*{\mglselementreset}[1]{%
13967   \ifKV@mglspresetlocal
13968   \glslocalreset{#1}%
13969   \else
13970   \glsreset{#1}%
13971   \fi
13972 }
```

glselementunset

```
13973 \newcommand*{\mglselementunset}[1]{%
13974   \ifKV@mglspresetlocal
13975   \glslocalunset{#1}%
13976   \else
13977   \glsunset{#1}%
13978   \fi
13979 }
```

\@mgl@resetall

```
13980 \newcommand*{\@mgl@resetall}{}
13981 \define@choicekey{mgl}{resetall}%
13982 [\@mgl@resetall@val\@mgl@resetall@nr]{false,true}[true]%
13983 {%
13984   \ifcase\@mgl@resetall@nr\relax
13985   \renewcommand*{\@mgl@resetall}{}%
13986   \or
13987   \renewcommand*{\@mgl@resetall}{%
13988     \@for\@gls@resetlabel:=\mglcurrentlist\do{\mglselementreset\@gls@resetlabel}}%
13989   \renewcommand*{\@mgl@unsetall}{}%
13990   \fi
13991 }
```

@mgl@resetmain

```
13992 \newcommand*{\@mgl@resetmain}{}
13993 \define@choicekey{mgl}{resetmain}
13994 [\@mgl@resetmain@val\@mgl@resetmain@nr]{false,true}[true]%
13995 {%
13996   \ifcase\@mgl@resetmain@nr\relax
13997   \renewcommand*{\@mgl@resetmain}{}%
13998   \or
13999   \renewcommand*{\@mgl@resetmain}{\mglselementreset\mglcurrentmainlabel}%
14000   \renewcommand*{\@mgl@unsetmain}{}%
14001   \fi
14002 }
```

gls@resetothers

```
14003 \newcommand*{\@mgl@resetothers}{}
14004 \define@choicekey{mgl}{resetothers}
14005 [\@mgl@resetothers@val\@mgl@resetothers@nr]{false,true}[true]%
```

```

14006 {%
14007 \ifcase\@mgl@resetothers@nr\relax
14008 \renewcommand*{\@mgl@resetothers}{}%
14009 \or
14010 \renewcommand*{\@mgl@resetothers}{%
14011 \@for\@gl@resetlabel:=\mglcurrentlist\do{%
14012 \ifx\@gl@resetlabel\mglcurrentmainlabel
14013 \else
14014 \mglselementreset\@gl@resetlabel
14015 \fi
14016 }%
14017 }%
14018 \renewcommand*{\@mgl@unsetothers}{}%
14019 \fi
14020 }

```

\@mgl@unsetall

```

14021 \newcommand*{\@mgl@unsetall}{%
14022 \define@choicekey{mgl}{unsetall}%
14023 [\@mgl@unsetall@val\@mgl@unsetall@nr]{false,true}[true]%
14024 {%
14025 \ifcase\@mgl@unsetall@nr\relax
14026 \renewcommand*{\@mgl@unsetall}{}%
14027 \or
14028 \renewcommand*{\@mgl@unsetall}{%
14029 \@for\@gl@unsetlabel:=\mglcurrentlist\do{\mglselementunset\@gl@unsetlabel}}%
14030 \renewcommand*{\@mgl@resetall}{}%
14031 \fi
14032 }

```

@mgl@unsetmain

```

14033 \newcommand*{\@mgl@unsetmain}{%
14034 \define@choicekey{mgl}{unsetmain}
14035 [\@mgl@unsetmain@val\@mgl@unsetmain@nr]{false,true}[true]%
14036 {%
14037 \ifcase\@mgl@unsetmain@nr\relax
14038 \renewcommand*{\@mgl@unsetmain}{}%
14039 \or
14040 \renewcommand*{\@mgl@unsetmain}{\mglselementunset\mglcurrentmainlabel}%
14041 \renewcommand*{\@mgl@resetmain}{}%
14042 \fi
14043 }

```

gl@unsetothers

```

14044 \newcommand*{\@mgl@unsetothers}{%
14045 \define@choicekey{mgl}{unsetothers}
14046 [\@mgl@unsetothers@val\@mgl@unsetothers@nr]{false,true}[true]%
14047 {%
14048 \ifcase\@mgl@unsetothers@nr\relax

```

```

14049 \renewcommand*{\@mgls@unsetothers}{}%
14050 \or
14051 \renewcommand*{\@mgls@unsetothers}{%
14052   \@for\@gls@unsetlabel:=\mglscurrentlist\do{%
14053     \ifx\@gls@unsetlabel\mglscurrentmainlabel
14054       \else
14055         \mglselementunset\@gls@unsetlabel
14056       \fi
14057     }%
14058   }%
14059 \renewcommand*{\@mgls@resetothers}{}%
14060 \fi
14061 }

```

setup@docurrent Set up the commands to determine whether or not to do the current element.

```

14062 \newcommand{\glsxtr@setup@docurrent}{%
  \mglscurrentlabel expands to the label of the current element. Should this element be
  skipped?
14063 \ifx\mglscurrentlabel\mglscurrentmainlabel
  Main element. Should it be skipped?
14064 \mglsisfirstuse
14065 {%
14066   \ifKV@glsxtrcombined@firstskipmain
14067     \let\@mgls@do@current@element\@gobble
14068   \else
14069     \let\@mgls@do@current@element\@firstofone
14070   \fi
14071 }%
14072 {%
14073   \ifKV@glsxtrcombined@usedskipmain
14074     \let\@mgls@do@current@element\@gobble
14075   \else
14076     \let\@mgls@do@current@element\@firstofone
14077   \fi
14078 }%
14079 \else
  Other element. Should it be skipped?
14080 \mglsisfirstuse
14081 {%
14082   \ifKV@glsxtrcombined@firstskipothers
14083     \let\@mgls@do@current@element\@gobble
14084   \else
14085     \let\@mgls@do@current@element\@firstofone
14086   \fi
14087 }%
14088 {%
14089   \ifKV@glsxtrcombined@usedskipothers

```

```

14090      \let\@mgls@do@current@element\@gobble
14091      \else
14092      \let\@mgls@do@current@element\@firstofone
14093      \fi
14094    }%
14095  \fi
14096 }

```

checklastelement If the last element is skipped, \mglsiflast needs adjusting. The first argument should be either first or used. The second argument is the multi-element label.

```

14097 \newcommand*{\glstr@mgls@checklastelement}[2]{%
14098   \ifbool{KV@glstrcombined@#1skipmain}%
14099   {%
14100     \ifbool{KV@glstrcombined@#1skipothers}%
14101     {%

```

This condition has already been checked for.

```

14102   }%
14103   {%

```

Main skipped. The last item will be the last other element.

```

14104     \ifnum\mglselementindex=\glstrmultilastotherindex{#2}\relax
14105     \let\mglsiflast\@firstoftwo
14106   \else
14107     \let\mglsiflast\@secondoftwo
14108   \fi
14109 }%
14110 }%
14111 {%

```

Main not skipped.

```

14112   \ifbool{KV@glstrcombined@#1skipothers}%
14113   {%

```

Others skipped. The main element is the only item.

```

14114     \ifnum\mglselementindex=\glstrmultimainindex{#2}\relax
14115     \let\mglsiflast\@firstoftwo
14116   \else
14117     \let\mglsiflast\@secondoftwo
14118   \fi
14119 }%
14120 {%

```

None skipped. This isn't the last element.

```

14121   \let\mglsiflast\@secondoftwo
14122 }%
14123 }%
14124 }

```

sWarnAllSkipped Warning if all elements are skipped. The first argument is the warning message, the second argument is the inserted content (final optional argument), the third command is the encapsulation command (which may be a hyperlink).

```

14125 \newcommand{\glstrmglsWarnAllSkipped}[3]{%
14126   \GlossariesExtraWarning{#1}%
14127   #3{#2}%
14128 }

```

@mgls@applyopts

```

14129 \newcommand*{\glstr@gls@applyopts}[1]{%
14130   \edef\@mgls@doptions{\noexpand\setkeys*{mgls}{\expandonce#1}}%
14131   \@mgls@doptions

```

Append any unknown options to all.

```

14132   \ifdefvoid\XKV@rm{{\eappto\@mgls@all{,\expandonce\XKV@rm}}}%

```

If setup key has been used, check for pre-option keys:

```

14133   \ifdefvoid\@mgls@setup
14134   {%
14135     {%
14136       \edef\@mgls@doptions{%
14137         \noexpand\setkeys*{glstrcombinedpreopts}{\expandonce\@mgls@setup}}%
14138       \mgls@disable@mgls@opts
14139       \@mgls@doptions
14140       \mgls@enable@mgls@opts

```

Save remaining setup options.

```

14141       \ifx\@mgls@setuptoptions\@empty
14142       \let\@mgls@setuptoptions\XKV@rm
14143       \else
14144       \eappto\@mgls@setuptoptions{,\expandonce\XKV@rm}%
14145       \fi
14146     }%

```

Apply gls unset/reset options.

```

14147   \@mgls@resetall
14148   \@mgls@unsetall
14149   \@mgls@resetmain
14150   \@mgls@unsetmain
14151   \@mgls@resetothers
14152   \@mgls@unsetothers

```

Disable.

```

14153   \let\@mgls@resetall\@empty
14154   \let\@mgls@resetmain\@empty
14155   \let\@mgls@resetothers\@empty
14156   \let\@mgls@unsetall\@empty
14157   \let\@mgls@unsetmain\@empty
14158   \let\@mgls@unsetothers\@empty

```


First use flags.

```
14159 \ifmglsused\mglscurrentmultilabel
14160 {\let\mglsisfirstuse\@secondoftwo}%
14161 {\let\mglsisfirstuse\@firstoftwo}%
14162 }
```

\@firstofthree

```
14163 \providecommand{\@firstofthree}[3]{#1}
```

\@secondofthree

```
14164 \providecommand{\@secondofthree}[3]{#2}
```

\@thirdofthree

```
14165 \providecommand{\@thirdofthree}[3]{#3}
```

The main internal command for referencing multi-entries:

sxtr@mgls@inner

`\glxsxtr@mgls@inner{<options>}{<label>}{<insert>}{<first cs>}{<not first cs>}{<main first cs>}{<main other cs>}`

```
14166 \newcommand*\glxsxtr@mgls@inner[7]{%
14167 \let\mglslastmainlabel\@empty
14168 \let\mglsiflastmainwasfirstuse\@firstoftwo
14169 \let\mglsiflastmainwasplural\@secondoftwo
14170 \let\mglsiflastmaincapscase\@firstofthree
14171 \let\mglsiflastmainwasskipped\@firstoftwo
14172 \bgroup
14173 \ifcsundef{@gls@combined@#2@main}%
14174 {%
14175 \glxsxtrundefaction{Multi entry ‘#2’ hasn’t been defined}%
14176 {You need to define ‘#2’ with \string\multiglossaryentry}%
14177 \gdef\@mgls@post@hookdefs{%
14178 \protected@edef\mglslastmultilabel{#2}%
14179 \let\mglswasfirstuse\@firstoftwo
14180 \let\mglslastcategory\@empty
14181 \let\mglsiflastelementwasskipped\@firstoftwo
14182 \let\mglsiflastelementwasfirstuse\@firstoftwo
14183 \let\mglsiflastelementwasplural\@secondoftwo
14184 \let\mglsiflastelementcapscase\@firstofthree
14185 \let\mglslastelementlabel\@empty
14186 \let\mgls@do@postlinkhook\relax
14187 }%
14188 }%
14189 {%
14190 \protected@edef\mglscurrentmultilabel{#2}%
```

```

14191 \letcs\mglscurrentmainlabel{@gls@combined@#2@main}%
14192 \letcs\mglscurrentlist{@gls@combined@#2@list}%
14193 \letcs\mglscurrentoptions{@gls@combined@#2@options}%

```

Initialise (may be changed if multiunset is present):

```

14194 \ifmglsused\mglscurrentmultilabel
14195 {\let\mglsisfirstuse\@secondoftwo}%
14196 {\let\mglsisfirstuse\@firstoftwo}%

```

Only obtain pre-option keys:

```

14197 \edef\@mgls@doptions{%
14198 \noexpand\setkeys*{glxstrcombinedpreopts}{\expandonce\mglscurrentoptions}}%
14199 \@mgls@doptions

```

Save remaining setup options.

```

14200 \let\@mgls@setupoptions\XKV@rm

```

Apply \mgls options.

```

14201 \mgls@disable@setup
14202 \ifdefvoid\@gls@combined@mglsopts
14203 {}%
14204 {\glxstr@mgl@applyopts\@gls@combined@mglsopts}%
14205 \mgls@enable@setup

```

Apply options provided in #1.

```

14206 \ifstrempy{#1}{\def\@mgls@options{#1}\glxstr@mgl@applyopts\@mgls@options}%

```

Check for attribute settings.

```

14207 \ifx\@gls@combined@category\empty

```

No category

```

14208 \else

```

Attribute options:

```

14209 \glshascategoryattribute{\@gls@combined@category}{multioptions}%
14210 {%
14211 \letcs\@mgls@attroptions{@glxstr@categoryattr@@\@gls@combined@category
14212 @multioptions}%

```

Only obtain pre-option keys:

```

14213 \let\@gls@combined@mglsopts\@empty
14214 \edef\@mgls@doptions{%
14215 \noexpand\setkeys*{glxstrcombinedpreopts}{\expandonce\@mgls@attroptions}}%
14216 \@mgls@doptions

```

Append remaining options:

```

14217 \eappto\@mgls@setupoptions{,\expandonce\XKV@rm}%
14218 \ifx\@gls@combined@mglsopts\@empty
14219 \else

```

mgls options found:

```

14220 \let\@mgls@setup\@empty
14221 \mgls@disable@setup
14222 \glxstr@mgl@applyopts\@gls@combined@mglsopts

```

```

14223         \mgl@enable@setup
14224     \fi
14225 }%
14226 {}%
14227 \fi

```

Apply setup options.

```

14228 \edef\@mgl@doptions{%
14229     \noexpand\setkeys{glsxtrcombined}{\expandonce\@mgl@setuptoptions}}%
14230 \@mgl@doptions

```

Provide local user-level access to category:

```

14231 \let\mgl@currentcategory\@gls@combined@category

```

Should the entire content be a hyperlink?

```

14232 \ifnum\@gls@combined@hyper=1\relax
14233     \def\@mgl@combinedlink{\@mgl@hyperlink{\mgl@currentmainlabel}}%
14234 \else
14235     \def\@mgl@combinedlink{\@firstofone}%
14236 \fi

```

Entire content encapsulator.

```

14237 \def\@gls@combined@encapsulator##1{%
14238     \@mgl@combinedlink{\csuse{\@gls@combined@textformat}{##1}}}%

```

Initialise.

```

14239 \let\@mgl@do@current@element\@firstofone

```

Check if all elements are being skipped.

```

14240 \mgl@isfirstuse
14241 {%
14242     \ifKV@glxtrcombined@firstskipmain
14243     \ifKV@glxtrcombined@firstskipothers

```

Just do the warning and insert. This will ignore the loop.

```

14244         \let\@gls@org@combined@encapsulator\@gls@combined@encapsulator
14245         \def\@gls@combined@encapsulator##1{%
14246             \glxtrmgl@WarnAllSkipped{All elements skipped for
14247                 first use of multi-entry '#2'}{#3}%
14248             {\@gls@org@combined@encapsulator}%
14249         }%
14250         \let\@mgl@do@current@element\@gobble
14251     \fi
14252 \fi
14253 }%
14254 {%
14255     \ifKV@glxtrcombined@usedskipmain
14256     \ifKV@glxtrcombined@usedskipothers

```

Just do the warning and insert. This will ignore the loop.

```

14257         \let\@gls@org@combined@encapsulator\@gls@combined@encapsulator
14258         \def\@gls@combined@encapsulator##1{%

```

```

14259         \glstrmglWarnAllSkipped{All elements skipped for
14260             subsequent use of multi-entry ‘#2’}{#3}%
14261         {\@gls@org@combined@encapsulator}%
14262     }%
14263     \let\@mgls@do@current@element\@gobble
14264 \fi
14265 \fi
14266 }%

```

Determine prefix and suffix.

```

14267 \mglsisfirstuse
14268 {%
14269     \let\mglscurrentprefix\@gls@combined@firstprefix
14270     \let\mglscurrentsuffix\@gls@combined@firstsuffix
14271 }%
14272 {%
14273     \let\mglscurrentprefix\@gls@combined@usedprefix
14274     \let\mglscurrentsuffix\@gls@combined@usedsuffix
14275 }%

```

Set up post-link hook used after current scope.

```

14276 \xdef\@mgls@post@hookdefs{%
14277     \noexpand\def\noexpand\mglslastmultilabel{\expandonce\mglscurrentmultilabel}%
14278     \noexpand\def\noexpand\mglslastcategory{\mglscurrentcategory}%
14279 }%
14280 \ifx\@mgls@do@current@element\@gobble
14281     \gappto\@mgls@post@hookdefs{%
14282         \let\mglsiflastelementskipped\@firstoftwo
14283         \let\mglslastelementlabel\@empty
14284         \let\mglsiflastelementwasfirstuse\@firstoftwo
14285         \let\mglsiflastelementwasplural\@secondoftwo
14286         \let\mglsiflastelementcapscase\@firstofthree
14287     }%
14288 \fi
14289 \mglsisfirstuse
14290 {%
14291     \gappto\@mgls@post@hookdefs{\let\mglswasfirstuse\@firstoftwo}%

```

Determine if the multi-entry post-link hook should be applied.

```

14292     \ifcase\@gls@combined@mpostlink@nr\relax
14293         mpostlink=false.
14293         \gappto\@mgls@post@hookdefs{\let\mgls@do@postlinkhook\relax}%
14294     \or
14295         mpostlink=true.
14295         \ifcase\@gls@combined@mpostlinkelement@nr\relax
14296             \gappto\@mgls@post@hookdefs{\let\mgls@do@postlinkhook\mglslastelementpostlinkhook}%
14297         \or
14298             \gappto\@mgls@post@hookdefs{\let\mgls@do@postlinkhook\mglslastmainpostlinkhook}%
14299         \or

```

```

14300         \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglScustompostlinkhook}%
14301     \fi
14302 \or
    mpostlink=firstonly.
14303     \ifcase\@glS@combined@mpostlinkelement@nr\relax
14304         \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglSlastelementpostlinkhook}%
14305     \or
14306         \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglSlastmainpostlinkhook}%
14307     \or
14308         \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglScustompostlinkhook}%
14309     \fi
14310 \or
    mpostlink=usedonly.
14311     \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\relax}%
14312 \fi
14313 }%
14314 {%
14315     \gappto\@mglS@post@hookdefs{\let\mglSwasfirstuse\@secondoftwo}%
    Determine if the multi-entry post-link hook should be applied.
14316     \ifcase\@glS@combined@mpostlink@nr\relax
        mpostlink=false.
14317         \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\relax}%
14318     \or
        mpostlink=true.
14319         \ifcase\@glS@combined@mpostlinkelement@nr\relax
14320             \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglSlastelementpostlinkhook}%
14321         \or
14322             \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglSlastmainpostlinkhook}%
14323         \or
14324             \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglScustompostlinkhook}%
14325         \fi
14326     \or
        mpostlink=firstonly.
14327         \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\relax}%
14328     \or
        mpostlink=usedonly.
14329         \ifcase\@glS@combined@mpostlinkelement@nr\relax
14330             \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglSlastelementpostlinkhook}%
14331         \or
14332             \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglSlastmainpostlinkhook}%
14333         \or
14334             \gappto\@mglS@post@hookdefs{\let\mglS@do@postlinkhook\mglScustompostlinkhook}%
14335         \fi
14336     \fi
14337 }%

```

Save current post-link hook.

```

14338 \let\mgls@org@postlinkhook\glspostlinkhook
    Prefix.
14339 \mglsprefix
    Initialise last element label (for \mglssuffix).
14340 \let\mglslastelementlabel\@empty
14341 \@gls@combined@encapsulator
14342 {%
    Save previous label.
14343 \def\@mgls@previouslabel{}%
14344 \mglselementindex=0\relax
14345 \@for\mglscurrentlabel:=\mglscurrentlist\do{%
14346 \advance\mglselementindex by 1\relax
14347 \glsxtr@setup@docurrent
    Is this the last element?
14348 \ifx\@xfor@nextelement\@nnil
14349 \let\mglsiflast\@firstoftwo
14350 \else
14351 \let\mglsiflast\@secondoftwo
    Are any elements being skipped?
14352 \mglsisfirstuse
14353 {%
14354 \glsxtr@mgls@checklastelement{first}{#2}%
14355 }%
14356 {%
14357 \glsxtr@mgls@checklastelement{used}{#2}%
14358 }%
14359 \fi
    Should the element post-link hook be used?
14360 \ifcase\@gls@combined@postlinks@nr\relax
    postlinks=none
14361 \let\glspostlinkhook\relax
14362 \or
    postlinks=all
14363 \let\glspostlinkhook\mgls@org@postlinkhook
14364 \or
    postlinks=notlast
14365 \mglsiflast
14366 {%
14367 \let\glspostlinkhook\relax
14368 }%
14369 {%
14370 \let\glspostlinkhook\mgls@org@postlinkhook
14371 }%
14372 \or

```

```

    postlinks=mainnotlast
14373     \ifx\mglscurrentlabel\mglscurrentmainlabel
14374         \mglciflast
14375         {%
14376             \let\glspostlinkhook\relax
14377         }%
14378         {%
14379             \let\glspostlinkhook\mglorg@postlinkhook
14380         }%
14381     \else
14382         \let\glspostlinkhook\relax
14383     \fi
14384 \or

    postlinks=mainonly
14385     \ifx\mglscurrentlabel\mglscurrentmainlabel
14386         \let\glspostlinkhook\mglorg@postlinkhook
14387     \else
14388         \let\glspostlinkhook\relax
14389     \fi
14390 \or

    postlinks=othernotlast
14391     \ifx\mglscurrentlabel\mglscurrentmainlabel
14392         \let\glspostlinkhook\relax
14393     \else
14394         \mglciflast
14395         {%
14396             \let\glspostlinkhook\relax
14397         }%
14398         {%
14399             \let\glspostlinkhook\mglorg@postlinkhook
14400         }%
14401     \fi
14402 \or

    postlinks=otheronly
14403     \ifx\mglscurrentlabel\mglscurrentmainlabel
14404         \let\glspostlinkhook\relax
14405     \else
14406         \let\glspostlinkhook\mglorg@postlinkhook
14407     \fi
14408 \fi

    Save the last element for the multi-entry post-link hook.
14409     \mglciflast
14410     {%
14411         \xappto\@mglorg@post@hookdefs{%
14412             \noexpand\def\noexpand\mglslastelementlabel
14413                 {\expandonce\mglscurrentlabel}}%
14414     }%

```

```

14415      {}%

Do current element:
14416      \@mgls@do@current@element
14417      {%

Pre element hook.
14418      \mglselementprehook

Is this the first use of the current element?
14419      \GlsXtrIfUnusedOrUndefined{\mglscurrentlabel}%
14420      {\let\@mgls@current@iffirstuse\@firstoftwo}%
14421      {\let\@mgls@current@iffirstuse\@secondoftwo}%
14422      \ifx\mglscurrentlabel\mglscurrentmainlabel

Main element. Location encap option:
14423      \edef\@mgls@current@options{format=\@gls@combined@encapmain}%

Indexing option:
14424      \ifcase\@gls@combined@indexmain
14425      \appto\@mgls@current@options{,noindex}%
14426      \or
14427      \appto\@mgls@current@options{,noindex=false}%
14428      \or
14429      \@mgls@current@iffirstuse
14430      {\appto\@mgls@current@options{,noindex=false}}%
14431      {\appto\@mgls@current@options{,noindex}}%
14432      \fi

Hyperlink option:
14433      \ifcase\@gls@combined@hyper\relax
14434      \appto\@mgls@current@options{,hyper=false}% none
14435      \or
14436      \appto\@mgls@current@options{,hyper=false}% allmain
14437      \or
14438      \eappto\@mgls@current@options{,\@mgls@hyper}% mainonly
14439      \or
14440      \eappto\@mgls@current@options{,\@mgls@hyper}% individual
14441      \or
14442      \appto\@mgls@current@options{,hyper=false}% otheronly
14443      \or
14444      \mglsisfirstuse
14445      {%
14446      \appto\@mgls@current@options{,hyper=false}% notmainfirst
14447      }%
14448      {%
14449      \eappto\@mgls@current@options{,\@mgls@hyper}% notmainfirst
14450      }%
14451      \or
14452      \eappto\@mgls@current@options{,\@mgls@hyper}% nototherfirst
14453      \or
14454      \mglsisfirstuse

```



```

14455      {%
14456      \appto\@mgl@current@options{,hyper=false}% notfirst
14457      }%
14458      {%
14459      \eappto\@mgl@current@options{,\@mgl@hyper}% notfirst
14460      }%
14461      \fi

```

Append all and then main:

```

14462      \eappto\@mgl@current@options{,\@mgl@all,\@mgl@main}%
14463      \else

```

Other element. Location encap option:

```

14464      \edef\@mgl@current@options{format=\@gls@combined@encapothers}%

```

Indexing option:

```

14465      \ifcase\@gls@combined@indexothers\relax
14466      \appto\@mgl@current@options{,noindex}%
14467      \or
14468      \appto\@mgl@current@options{,noindex=false}%
14469      \or
14470      \@mgl@current@iffirstuse
14471      {\appto\@mgl@current@options{,noindex=false}}%
14472      {\appto\@mgl@current@options{,noindex}}%
14473      \fi

```

Hyperlink option:

```

14474      \ifcase\@gls@combined@hyper\relax
14475      \appto\@mgl@current@options{,hyper=false}% none
14476      \or
14477      \appto\@mgl@current@options{,hyper=false}% allmain
14478      \or
14479      \appto\@mgl@current@options{,hyper=false}% mainonly
14480      \or
14481      \eappto\@mgl@current@options{,\@mgl@hyper}% individual
14482      \or
14483      \eappto\@mgl@current@options{,\@mgl@hyper}% otheronly
14484      \or
14485      \eappto\@mgl@current@options{,\@mgl@hyper}% notmainfirst
14486      \or
14487      \mgl@sisfirstuse
14488      {%
14489      \appto\@mgl@current@options{,hyper=false}% nototherfirst
14490      }%
14491      {%
14492      \eappto\@mgl@current@options{,\@mgl@hyper}% nototherfirst
14493      }%
14494      \or
14495      \mgl@sisfirstuse
14496      {%
14497      \appto\@mgl@current@options{,hyper=false}% notfirst

```

```

14498         }%
14499         {%
14500         \eappto\@mglscurrent@options{,\@mglshyper}% notfirst
14501         }%
14502     \fi

```

Append all and then others:

```

14503         \eappto\@mglscurrent@options{,\@mglscall,\@mglsothers}%
14504     \fi

```

Is this the first element?

```

14505     \ifx\@mglspreviouslabel\empty
14506     \ifx\mglscurrentlabel\mglscurrentmainlabel
14507         \let\@mglscs#6\relax
14508     \else
14509         \let\@mglscs#4\relax
14510     \fi
14511 \else

```

Not the first element so add separator.

```

14512     \@mglsprevious@iffirstuse
14513     {%
14514         \@mglscurrent@iffirstuse
14515         {\glscombinedfirstsepfirst{\@mglspreviouslabel}{\mglscurrentlabel}}%
14516         {\glscombinedfirstsep{\@mglspreviouslabel}{\mglscurrentlabel}}%
14517     }%
14518     {%
14519         \@mglscurrent@iffirstuse
14520         {\glscombinedsepfirst{\@mglspreviouslabel}{\mglscurrentlabel}}%
14521         {\glscombinedsep{\@mglspreviouslabel}{\mglscurrentlabel}}%
14522     }%
14523     \ifx\mglscurrentlabel\mglscurrentmainlabel
14524         \let\@mglscs#7\relax
14525     \else
14526         \let\@mglscs#5\relax
14527     \fi
14528 \fi

```

Is this the last element?

```

14529     \mglusiflast
14530     {\xandafter\@mglscs\xandafter{\@mglscurrent@options}{\mglscurrentlabel}[\#3]}%
14531     {\xandafter\@mglscs\xandafter{\@mglscurrent@options}{\mglscurrentlabel}[]}%

```

Is this the main element? If so, save information for post-link hook.

```

14532     \ifx\mglscurrentlabel\mglscurrentmainlabel
14533         \xappto\@mglspost@hookdefs{%
14534             \noexpand\def\noexpand\mglslastmainlabel
14535                 {\expandonce\mglscurrentmainlabel}%
14536         }%
14537     \glxtrifwasfirstuse
14538     {%

```

```

14539     \gappto\@mgls@post@hookdefs{\let\mglsiflastmainwasfirstuse\@firstoftwo}%
14540 }%
14541 {%
14542     \gappto\@mgls@post@hookdefs{\let\mglsiflastmainwasfirstuse\@secondoftwo}%
14543 }%
14544 \glsifplural
14545 {%
14546     \gappto\@mgls@post@hookdefs{\let\mglsiflastmainwasplural\@firstoftwo}%
14547 }%
14548 {%
14549     \gappto\@mgls@post@hookdefs{\let\mglsiflastmainwasplural\@secondoftwo}%
14550 }%
14551 \glsifcaps
14552 {%
14553     \gappto\@mgls@post@hookdefs{%
14554         \let\mglsiflastmaincaps\@firstoftwo
14555     }%
14556 }%
14557 {%
14558     \gappto\@mgls@post@hookdefs{%
14559         \let\mglsiflastmaincaps\@secondoftwo
14560     }%
14561 }%
14562 {%
14563     \gappto\@mgls@post@hookdefs{%
14564         \let\mglsiflastmaincaps\@thirdoftwo
14565     }%
14566 }%
14567 \fi
14568 \let\@mgls@previouslabel\mglscurrentlabel
14569 \let\@mgls@previous@iffirstuse\@mgls@current@iffirstuse
14570 }%

```

Post element hook.

```

14571     \mgls@element@posthook
14572 }%
14573 \ifx\mgls@lastmainlabel\@empty
14574     \gappto\@mgls@post@hookdefs{\let\mglsiflastmainskipped\@firstoftwo}%
14575 \else
14576     \gappto\@mgls@post@hookdefs{\let\mglsiflastmainskipped\@secondoftwo}%
14577 \fi

```

Encapsulator may introduce grouping so check here.

```

14578     \ifx\@mgls@do@current@element\@gobble
14579     \gappto\@mgls@post@hookdefs{\let\mglsiflastelementskipped\@firstoftwo}%
14580 \else
14581     \gappto\@mgls@post@hookdefs{\let\mglsiflastelementskipped\@secondoftwo}%
14582 \fi
14583 \glsxtrifwasfirstuse
14584 {%

```

```

14585 \gappto\@mgls@post@hookdefs{\let\mglsiflastelementwasfirstuse\@firstoftwo}%
14586 }%
14587 {%
14588 \gappto\@mgls@post@hookdefs{\let\mglsiflastelementwasfirstuse\@secondoftwo}%
14589 }%
14590 \glsifplural
14591 {%
14592 \gappto\@mgls@post@hookdefs{\let\mglsiflastelementwasplural\@firstoftwo}%
14593 }%
14594 {%
14595 \gappto\@mgls@post@hookdefs{\let\mglsiflastelementwasplural\@secondoftwo}%
14596 }%
14597 \gls caps case
14598 {%
14599 \gappto\@mgls@post@hookdefs{%
14600 \let\mglsiflastelementcaps case\@firstofthree
14601 }%
14602 }%
14603 {%
14604 \gappto\@mgls@post@hookdefs{%
14605 \let\mglsiflastelementcaps case\@secondofthree
14606 }%
14607 }%
14608 {%
14609 \gappto\@mgls@post@hookdefs{%
14610 \let\mglsiflastelementcaps case\@thirdofthree
14611 }%
14612 }%
14613 }%

```

Suffix needs post-link hook commands.

```

14614 \@mgls@post@hookdefs
14615 \mgls suffix

```

Unset multi-entry first use flag after current scope.

```

14616 \ifcase\@mgls@unsetaction\relax
14617 \xappto\@mgls@post@hookdefs{%
14618 \noexpand\mglsunset{\expandonce\mgls current multilabel}}%
14619 \or
14620 \xappto\@mgls@post@hookdefs{%
14621 \noexpand\mglslocalunset{\expandonce\mgls current multilabel}}%
14622 \fi
14623 }%
14624 \glsxtrmgls write{#2}%
14625 \egroup
14626 \@mgls@post@hookdefs
14627 \mgls@do@postlinkhook
14628 }

```

tompostlinkhook

```

14629 \newcommand*{\mglscustompostlinkhook}{}

entpostlinkhook
14630 \newcommand*{\mglslastelementpostlinkhook}{%
14631 \let\glsextrifwasfirstuse\mglusiflastelementwasfirstuse
14632 \let\glusifplural\mglusiflastelementwasplural
14633 \let\glscapscase\mglusiflastelementcapscase
14634 \let\glslabel\mglslastelementlabel
14635 \glspostlinkhook
14636 }

ainpostlinkhook
14637 \newcommand*{\mglslastmainpostlinkhook}{%
14638 \let\glsextrifwasfirstuse\mglusiflastmainwasfirstuse
14639 \let\glusifplural\mglusiflastmainwasplural
14640 \let\glscapscase\mglusiflastmaincapscase
14641 \let\glslabel\mglslastmainlabel
14642 \glspostlinkhook
14643 }

fcategoryprefix
14644 \newcommand*{\mglsdefcategoryprefix}[2]{%
14645 \csdef{mglsprefix@#1}{#2}%
14646 }

scategoryprefix
14647 \newcommand*{\mglshascategoryprefix}[3]{%
14648 \ifcsdef{mglsprefix@#1}{#2}{#3}%
14649 }

ecategoryprefix
14650 \newcommand*{\mglsecategoryprefix}[1]{%
14651 \csuse{mglsprefix@#1}%
14652 }

\mglsprefix
14653 \newcommand*{\mglsprefix}{%
14654 \ifdefempty\mglscurrentcategory
14655 {\mglscurrentprefix}%
14656 {%
14657 \mglshascategoryprefix{\mglscurrentcategory}%
14658 {\mglsecategoryprefix{\mglscurrentcategory}}%
14659 {\mglscurrentprefix}%
14660 }%
14661 }

fcategorysuffix
14662 \newcommand*{\mglsdefcategorysuffix}[2]{%
14663 \csdef{mglssuffix@#1}{#2}%
14664 }

```

scategorysuffix

```
14665 \newcommand*{\mglshascategorysuffix}[3]{%
14666   \ifcsdef{mglssuffix@#1}{#2}{#3}%
14667 }
```

ecategorysuffix

```
14668 \newcommand*{\mglsecategorysuffix}[1]{%
14669   \csuse{mglssuffix@#1}%
14670 }
```

\mglssuffix

```
14671 \newcommand*{\mglssuffix}{%
14672   \ifdefempty\mglscurrentcategory
14673   {\ifdefempty\mglscurrentsuffix}{\space(\mglscurrentsuffix)}}%
14674   {%
14675     \mglshascategorysuffix\mglscurrentcategory}%
14676     {\mglsecategorysuffix\mglscurrentcategory}}%
14677     {\ifdefempty\mglscurrentsuffix}{\space(\mglscurrentsuffix)}}%
14678   }%
14679 }
```

selementprehook

```
14680 \newcommand*{\mglselementprehook}{}
```

elementposthook

```
14681 \newcommand*{\mglselementposthook}{}
```

Separators.

\glscombinedsep Separator between two elements that have been marked as used. This takes the two element labels as arguments.

```
14682 \newcommand*{\glscombinedsep}[2]{%
14683   \glsattribute{#1}{combinedsep}%
14684   {\glsgetattribute{#1}{combinedsep}}%
14685   { }%
14686 }
```

edfirstsepfirsr Separator following and preceding a first use.

```
14687 \newcommand*{\glscombinedfirstsepfirsr}[2]{%
14688   \glsattribute{#1}{combinedfirstsepfirsr}%
14689   {\glsgetattribute{#1}{combinedfirstsepfirsr}}%
14690   {\glscombinedsep{#1}{#2}}%
14691 }
```

ombinedfirstsep Separator following a first use.

```
14692 \newcommand*{\glscombinedfirstsep}[2]{%
14693   \glsattribute{#1}{combinedfirstsep}%
14694   {\glsgetattribute{#1}{combinedfirstsep}}%
14695   {\glscombinedsep{#1}{#2}}%
14696 }
```

combinedsepfirst Separator preceding a first use.

```
14697 \newcommand*{\glscmbinedsepfirst}[2]{%
14698   \glshasattribute{#1}{combinedsepfirst}%
14699   {\glsggetattribute{#1}{combinedsepfirst}}%
14700   {\glscmbinedsep{#1}{#2}}%
14701 }
```

combinedsepabbrvnbs Provide shortcut for using non-breakable space following an abbreviation that has already been used.

```
14702 \newcommand*{\glissetcombinedsepabbrvnbs}{%
14703   \renewcommand*{\glscmbinedsep}[2]{%
14704     \glshasattribute{##1}{combinedsep}%
14705     {\glsggetattribute{##1}{combinedsep}}%
14706     {\ifhasshort{##1}{~}{ }}%
14707   }%
14708   \renewcommand*{\glscmbinedsepfirst}[2]{%
14709     \glshasattribute{##1}{combinedsepfirst}%
14710     {\glsggetattribute{##1}{combinedsepfirst}}%
14711     {\ifhasshort{##1}{~}{ }}%
14712   }%
14713   \renewcommand*{\glscmbinedfirstsep}[2]{%
14714     \glshasattribute{##1}{combinedfirstsep}%
14715     {\glsggetattribute{##1}{combinedfirstsep}}%
14716     { }%
14717   }%
14718   \renewcommand*{\glscmbinedfirstsepfirst}[2]{%
14719     \glshasattribute{##1}{combinedfirstsepfirst}%
14720     {\glsggetattribute{##1}{combinedfirstsepfirst}}%
14721     { }%
14722   }%
14723 }
```

combinedsepabbrvnone Provide shortcut for using nothing if either on next use are abbreviations (otherwise use space).

```
14724 \newcommand*{\glissetcombinedsepabbrvnone}{%
14725   \renewcommand*{\glscmbinedsep}[2]{%
14726     \glshasattribute{##1}{combinedsep}%
14727     {\glsggetattribute{##1}{combinedsep}}%
14728     {\ifhasshort{##1}{~}{\ifhasshort{##2}{~}{ }}}%
14729   }%
14730   \renewcommand*{\glscmbinedsepfirst}[2]{%
14731     \glshasattribute{##1}{combinedsepfirst}%
14732     {\glsggetattribute{##1}{combinedsepfirst}}%
14733     {\ifhasshort{##1}{~}{ }}%
14734   }%
14735   \renewcommand*{\glscmbinedfirstsep}[2]{%
14736     \glshasattribute{##1}{combinedfirstsep}%
14737     {\glsggetattribute{##1}{combinedfirstsep}}%
14738     {\ifhasshort{##2}{~}{ }}%
14739   }%
14740 }
```

```

14739 }%
14740 \renewcommand*{\glscombinedfirstsepfirst}[2]{%
14741   \glsattribute{##1}{combinedfirstsepfirst}%
14742   {\glsgetattribute{##1}{combinedfirstsepfirst}}%
14743   { }%
14744 }%
14745 }

```

mbinedsepnarrow Measures both.

```

14746 \newcommand*{\glssetcombinedsepnarrow}[2]{%
14747   \renewcommand*{\glscombinedsep}[2]{%
14748     \glsattribute{##1}{combinedsep}%
14749     {\glsgetattribute{##1}{combinedsep}}%
14750     {%
14751       \ifhasshort{##1}%
14752         {\settowidth{\dimen@}{\glsentryshort{##1}}}%
14753         {\settowidth{\dimen@}{\glsentrytext{##1}}}%
14754         \ifdim\dimen@<#1\relax
14755           #2%
14756         \else
14757           \ifhasshort{##2}%
14758             {\settowidth{\dimen@}{\glsentryshort{##2}}}%
14759             {\settowidth{\dimen@}{\glsentrytext{##2}}}%
14760             \ifdim\dimen@<#1\relax
14761               #2%
14762             \else
14763               \space
14764             \fi
14765           \fi
14766         }%
14767     }%
14768   \renewcommand*{\glscombinedsepfirst}[2]{%
14769     \glsattribute{##1}{combinedsepfirst}%
14770     {\glsgetattribute{##1}{combinedsepfirst}}%
14771     {%
14772       \ifhasshort{##1}%
14773         {\settowidth{\dimen@}{\glsentryshort{##1}}}%
14774         {\settowidth{\dimen@}{\glsentrytext{##1}}}%
14775         \ifdim\dimen@<#1\relax
14776           #2%
14777         \else
14778           \ifhaslong{##2}%
14779             {\settowidth{\dimen@}{\glsentrylong{##2}}}%
14780             {\settowidth{\dimen@}{\glsentryfirst{##2}}}%
14781             \ifdim\dimen@<#1\relax
14782               #2%
14783             \else
14784               \space
14785             \fi

```



```

14786   \fi
14787 }%
14788 }%
14789 \renewcommand*{\glscombinedfirstsep}[2]{%
14790   \glsattribute{##1}{combinedfirstsep}%
14791   {\glsgetattribute{##1}{combinedfirstsep}}%
14792   {%
14793     \ifhaslong{##1}%
14794     {\settowidth{\dimen@}{\glsentrylong{##1}}}%
14795     {\settowidth{\dimen@}{\glsentryfirst{##1}}}%
14796     \ifdim\dimen@<#1\relax
14797       #2%
14798     \else
14799       \ifhasshort{##2}%
14800       {\settowidth{\dimen@}{\glsentryshort{##2}}}%
14801       {\settowidth{\dimen@}{\glsentrytext{##2}}}%
14802       \ifdim\dimen@<#1\relax
14803         #2%
14804       \else
14805         \space
14806       \fi
14807     \fi
14808   }%
14809 }%
14810 \renewcommand*{\glscombinedfirstsepfirst}[2]{%
14811   \glsattribute{##1}{combinedfirstsepfirst}%
14812   {\glsgetattribute{##1}{combinedfirstsepfirst}}%
14813   {%
14814     \ifhaslong{##1}%
14815     {\settowidth{\dimen@}{\glsentrylong{##1}}}%
14816     {\settowidth{\dimen@}{\glsentryfirst{##1}}}%
14817     \ifdim\dimen@<#1\relax
14818       #2%
14819     \else
14820       \ifhaslong{##2}%
14821       {\settowidth{\dimen@}{\glsentrylong{##2}}}%
14822       {\settowidth{\dimen@}{\glsentryfirst{##2}}}%
14823       \ifdim\dimen@<#1\relax
14824         #2%
14825       \else
14826         \space
14827       \fi
14828     \fi
14829   }%
14830 }%
14831 }

```

lsxtr@mglswrite Write information to the aux file for bib2gls to pick up, but only need to do it once per label since it only indicates which multi-entry has been referenced without any additional

information.

```
14832 \newcommand{\glxtrmglswrite}[1]{%
14833   \ifx\@glxtr@record@setting\@glxtr@record@setting@off
14834   \else
14835     \protected@edef\@glxtr@mglslabel{#1}%
14836     \ifdef\@glxtr@mgl sreflist
14837     {%
14838       \expandafter\DTLifinlist\expandafter{\@glxtr@mglslabel}%
14839       {\@glxtr@mgl sreflist}{}%
14840       {%
14841         \xappto\@glxtr@mgl sreflist{,\expandonce\@glxtr@mglslabel}%
14842         \if@mgl s@writesepraterefs
14843           \protected@write\@auxout{}{\string\@glxtr@mgl srefs{#1}}%
14844         \fi
14845       }%
14846     }%
14847     {%
14848       \global\let\@glxtr@mgl sreflist\@glxtr@mglslabel
14849       \if@mgl s@writesepraterefs
14850         \protected@write\@auxout{}{\string\@glxtr@mgl srefs{#1}}%
14851       \else
14852         \AtEndDocument{\immediate\protected@write\@auxout{}%
14853           {\string\@glxtr@mgl srefs{\@glxtr@mgl sreflist}}}%
14854       \fi
14855       \@mgl s@disable@writeseprateref@cond
14856     }%
14857   \fi
14858 }
```

glxtr@mgl srefs

```
14859 \newcommand{\@glxtr@mgl srefs}[1]{}
```

iteseparaterefs If this conditional is changed, it must be done before the first instance of any \mgl s-like command.

```
14860 \newif\if@mgl s@writesepraterefs \@mgl s@writesepraterefsfalse
```

eparateRefsTrue

```
14861 \newcommand{\mgl sWriteSeparateRefsTrue}{\global\@mgl s@writesepraterefstrue}
```

parateRefsFalse

```
14862 \newcommand{\mgl sWriteSeparateRefsFalse}{\global\@mgl s@writesepraterefsfalse}
```

eparateref@cond

```
14863 \newcommand*{\@mgl s@disable@writeseprateref@cond}{%
14864   \gdef\mgl sWriteSeparateRefsTrue{\PackageError{glossaries-extra}%
14865     {Too late to use \string\mgl sWriteSeparateRefsTrue}%
14866     {\string\mgl sWriteSeparateRefsTrue\space can only be used before
14867     the first instance of any \string\mgl s-like command}}%
```

```

14868 \gdef\mglsWriteSeparateRefsFalse{\PackageError{glossaries-extra}%
14869 {Too late to use \string\mglsWriteSeparateRefsFalse}%
14870 {\string\mglsWriteSeparateRefsFalse\space can only be used before
14871 the first instance of any \string\mgls-like command}}%
14872 }

```

\glxtr@newmgls

```

14873 \newcommand{\glxtr@newmgls}[5]{%
14874 \edef\@glxtr@newmgls@do{%
14875 \noexpand\newrobustcmd*{\expandonce{\csname #1\endcsname}}%
14876 {\noexpand\@glxtr@hyp@opt\expandonce{\csname ns@glxtr@#1\endcsname}}%
14877 \noexpand\newcommand*{\expandonce{\csname ns@glxtr@#1\endcsname}}[2][{%
14878 \noexpand\new@ifnextchar[%
14879 {\expandonce{\csname glxtr@#1\endcsname}}{####1}{####2}}%
14880 {\expandonce{\csname glxtr@#1\endcsname}}{####1}{####2}}[]}%
14881 }%
14882 \noexpand\def\expandonce{\csname glxtr@#1\endcsname}####1####2[####3]{%
14883 \noexpand\def\noexpand\glxtrcurrentmglsname{#1}%
14884 \noexpand\glxtr@mgl@inner{####1}{####2}{####3}%
14885 {\noexpand#2}{\noexpand#3}{\noexpand#4}{\noexpand#5}%
14886 }%
14887 }%
14888 \@glxtr@newmgls@do
14889 \ifx\@glxtr@record@setting\@glxtr@record@setting@off
14890 \else

```

Provide a way for bib2gls to recognise the command (this will make it easier to add extra commands without having to modify bib2gls).

```

14891 \ifdef\@glxtr@mglslikelist
14892 {\xappto\@glxtr@mglslikelist{,#1}}%
14893 {%
14894 \gdef\@glxtr@mglslikelist{#1}%
14895 \AtEndDocument{\immediate\protected@write\@auxout{%
14896 {\string\@glxtr@mglslike{\@glxtr@mglslikelist}}}%
14897 }%
14898 \fi
14899 }

```

glxtr@mglslike

```

14900 \newcommand*{\@glxtr@mglslike}[1]{%

```

GlsXtrMglsOrGls

```

\GlsXtrMglsOrGls{<mgl s cs>}{<gls cs>}<modifier>[<options>]{<label>}
[<insert>]

```

```

14901 \newcommand*{\GlsXtrMglsOrGls}[2]{%

```

```

14902 \def\@glsxtr@mgl@s@or@gls@mcs{#1}%
14903 \def\@glsxtr@mgl@s@or@gls@gcs{#2}%
14904 \@ifstar{\s@GlsXtrMgl@sOrGls}%
14905 {%
14906 \@ifnextchar+(\PLUS\@firstoftwo{\p@GlsXtrMgl@sOrGls}}%
14907 {%
14908 \ifdefempty\@gls@alt@hyp@opt@char\@GlsXtrMgl@sOrGls\alt@GlsXtrMgl@sOrGls
14909 }%
14910 }%
14911 }

```

GlsXtrMgl@sOrGls

```

14912 \newcommand*{\alt@GlsXtrMgl@sOrGls}{
14913 \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
14914 {\@firstoftwo{\alt@GlsXtrMgl@sOrGls}}{\@GlsXtrMgl@sOrGls}%
14915 }

```

GlsXtrMgl@sOrGls

```

14916 \newcommand*{\@GlsXtrMgl@sOrGls}[2][{}]{%
14917 \glsxtrifmulti{#2}%
14918 {\@glsxtr@mgl@s@or@gls@mcs[#1]{#2}}%
14919 {\@glsxtr@mgl@s@or@gls@gcs[#1]{#2}}%
14920 }

```

GlsXtrMgl@sOrGls

```

14921 \newcommand*{\s@GlsXtrMgl@sOrGls}[2][{}]{%
14922 \glsxtrifmulti{#2}%
14923 {\@glsxtr@mgl@s@or@gls@mcs*[#1]{#2}}%
14924 {\@glsxtr@mgl@s@or@gls@gcs*[#1]{#2}}%
14925 }

```

GlsXtrMgl@sOrGls

```

14926 \newcommand*{\p@GlsXtrMgl@sOrGls}[2][{}]{%
14927 \glsxtrifmulti{#2}%
14928 {\@glsxtr@mgl@s@or@gls@mcs+[#1]{#2}}%
14929 {\@glsxtr@mgl@s@or@gls@gcs+[#1]{#2}}%
14930 }

```

GlsXtrMgl@sOrGls

```

14931 \newcommand*{\@alt@GlsXtrMgl@sOrGls}[2][{}]{%
14932 \glsxtrifmulti{#2}%
14933 {\expandafter\@glsxtr@mgl@s@or@gls@mcs\@gls@alt@hyp@opt@char[#1]{#2}}%
14934 {\expandafter\@glsxtr@mgl@s@or@gls@gcs\@gls@alt@hyp@opt@char[#1]{#2}}%
14935 }

```

\mgl@s

`\mgls[<options>]{<label>}[<insert>]`

Use `\gls` for all elements.

14936 `\glsxtr@newmgls{mgls}{\@gls@}{\@gls@}{\@gls@}{\@gls@}%`

`\mglspl`

`\mglspl[<options>]{<label>}[<insert>]`

Use `\glspl` for all elements.

14937 `\glsxtr@newmgls{mglspl}{\@glspl@}{\@glspl@}{\@glspl@}{\@glspl@}%`

`\mglsmainpl`

`\mglsmainpl[<options>]{<label>}[<insert>]`

Only use `\glspl` for the main element, otherwise use `\gls`.

14938 `\glsxtr@newmgls{mglsmainpl}{\@gls@}{\@gls@}{\@glspl@}{\@glspl@}%`

`\Mgls`

`\Mgls[<options>]{<label>}[<insert>]`

Use `\Gls` for first element and `\gls` for others.

14939 `\glsxtr@newmgls{Mgls}{\@Gls@}{\@gls@}{\@Gls@}{\@gls@}%`

`\Mglspl`

`\Mglspl[<options>]{<label>}[<insert>]`

Use `\Glspl` for first element and `\glspl` for others.

14940 `\glsxtr@newmgls{Mglspl}{\@Glspl@}{\@glspl@}{\@Glspl@}{\@glspl@}%`

`\Mglsmainpl`

`\Mglsmainpl[<options>]{<label>}[<insert>]`

Upper case the first element, no case change for others. Use plural for the main element only.

14941 `\glsxtr@newmgls{Mglsmainpl}{\@Gls@}{\@gls@}{\@Glspl@}{\@glspl@}%`

\MGlS

`\MGlS[<options>]{<label>}[<insert>]`

Use \GlS for all elements.

14942 \glSxtr@newmgls{MGlS}{\@GlS@}{\@GlS@}{\@GlS@}{\@GlS@}%

\MGlSpl

`\MGlSpl[<options>]{<label>}[<insert>]`

Use \GlSpl for all elements.

14943 \glSxtr@newmgls{MGlSpl}{\@GlSpl@}{\@GlSpl@}{\@GlSpl@}{\@GlSpl@}%

\MGlSmainpl

`\MGlSmainpl[<options>]{<label>}[<insert>]`

Start all elements with upper case. Only use plural for main element.

14944 \glSxtr@newmgls{MGlSmainpl}{\@GlS@}{\@GlS@}{\@GlSpl@}{\@GlSpl@}%

\MGLS

`\MGLS[<options>]{<label>}[<insert>]`

Use \GLS for all elements.

14945 \glSxtr@newmgls{MGLS}{\@GLS@}{\@GLS@}{\@GLS@}{\@GLS@}%

\MGLSpl

`\MGLSpl[<options>]{<label>}[<insert>]`

Use \GLSpl for all elements.

14946 \glSxtr@newmgls{MGLSpl}{\@GLSpl@}{\@GLSpl@}{\@GLSpl@}{\@GLSpl@}%

\MGLSmainpl

`\MGLSmainpl[<options>]{<label>}[<insert>]`

Upper case all elements. Only use plural for main element.

14947 \glSxtr@newmgls{MGLSmainpl}{\@GLS@}{\@GLS@}{\@GLSpl@}{\@GLSpl@}%

@glslongortext@

```
14948 \def\@glslongortext#1#2[#3]{%
14949   \ifglshaslong{#2}{\@glxtrlong{#1}{#2}[#3]}\@glstext@{#1}{#2}[#3]}%
14950 }
```

glsshortortext@

```
14951 \def\@glsshortortext#1#2[#3]{%
14952   \ifglshasshort{#2}{\@glxtrshort{#1}{#2}[#3]}\@glstext@{#1}{#2}[#3]}%
14953 }
```

glsfullorfirst@

```
14954 \def\@glsfullorfirst#1#2[#3]{%
14955   \ifglshasshort{#2}{\@glxtr@full{#1}{#2}[#3]}\@glsfirst@{#1}{#2}[#3]}%
14956 }
```

@Glslongortext@

```
14957 \def\@Glslongortext#1#2[#3]{%
14958   \ifglshaslong{#2}{\@Glsxtrlong{#1}{#2}[#3]}\@Glstext@{#1}{#2}[#3]}%
14959 }
```

Glsshortortext@

```
14960 \def\@Glsshortortext#1#2[#3]{%
14961   \ifglshasshort{#2}{\@Glsxtrshort{#1}{#2}[#3]}\@Glstext@{#1}{#2}[#3]}%
14962 }
```

Glsfullorfirst@

```
14963 \def\@Glsfullorfirst#1#2[#3]{%
14964   \ifglshasshort{#2}{\@Glsxtr@full{#1}{#2}[#3]}\@Glsfirst@{#1}{#2}[#3]}%
14965 }
```

\mglsshort

`\mglsshort[<options>]{<label>}[<insert>]`

Use short or text for all elements.

```
14966 \glxtr@newmgls{mglsshort}%
14967 {\@glsshortortext}\@glsshortortext{\@glsshortortext}\@glsshortortext}%
```

\mglslong

`\mglslong[<options>]{<label>}[<insert>]`

Use long or text for all elements.

```
14968 \glxtr@newmgls{mglslong}%
14969 {\@glslongortext}\@glslongortext{\@glslongortext}\@glslongortext}%
```

`\mglsfull`

`\mglsfull[<options>]{<label>}[<insert>]`

Use full or first for all elements.

14970 `\glstr@newmgls{mglsfull}%`

14971 `{\@glssfullorfirst}{\@glssfullorfirst}{\@glssfullorfirst}{\@glssfullorfirst}%`

`\Mglsshort`

`\Mglsshort[<options>]{<label>}[<insert>]`

Use short or text for all elements with initial cap on first element.

14972 `\glstr@newmgls{Mglsshort}%`

14973 `{\@Glsshortortext}{\@glssshortortext}{\@Glsshortortext}{\@glssshortortext}%`

`\Mglslong`

`\Mglslong[<options>]{<label>}[<insert>]`

Use long or text for all elements with initial cap on first element.

14974 `\glstr@newmgls{Mglslong}%`

14975 `{\@Glslongortext}{\@glslongortext}{\@Glslongortext}{\@glslongortext}%`

`\Mglsfull`

`\Mglsfull[<options>]{<label>}[<insert>]`

Use full or first for all elements with initial cap on first element.

14976 `\glstr@newmgls{Mglsfull}%`

14977 `{\@Glsfullorfirst}{\@glssfullorfirst}{\@Glsfullorfirst}{\@glssfullorfirst}%`

`\mglsname`

`\mglsname[<options>]{<label>}[<insert>]`

Use name for all elements.

14978 `\glstr@newmgls{mglsname}%`

14979 `{\@glsname@}{\@glsname@}{\@glsname@}{\@glsname@}%`

\Mglsname

`\Mglsname[<options>]{<label>}[<insert>]`

Use name for all elements with initial cap on first element.

```
14980 \glstr@newmgl{s}{Mglsname}%
14981 {\@Glsname@}{\@Glsname@}{\@Glsname@}{\@Glsname@}%
```

\MGlsname

`\MGlsname[<options>]{<label>}[<insert>]`

Use name for all elements with initial cap on all elements.

```
14982 \glstr@newmgl{s}{MGlsname}%
14983 {\@Glsname@}{\@Glsname@}{\@Glsname@}{\@Glsname@}%
```

@glssymbolorgls

```
14984 \def\@glssymbolorgls#1#2[#3]{%
14985   \ifglshassymbol{#2}{\@glssymbol@{#1}{#2}[#3]}\@gl{s@{#1}{#2}[#3]}%
14986 }
```

@glssymbolorGls

```
14987 \def\@glssymbolorGls#1#2[#3]{%
14988   \ifglshassymbol{#2}{\@glssymbol@{#1}{#2}[#3]}\@gl{S@{#1}{#2}[#3]}%
14989 }
```

\mglssymbol

`\mglssymbol[<options>]{<label>}[<insert>]`

Use \glssymbol if the symbol key is set otherwise use \gls.

```
14990 \glstr@newmgl{s}{mglssymbol}%
14991 {\@glssymbolorgls}{\@glssymbolorgls}{\@glssymbolorgls}{\@glssymbolorgls}%
```

\Mglssymbol

`\Mglssymbol[<options>]{<label>}[<insert>]`

As above but initial the first element if it's not a symbol.

```
14992 \glstr@newmgl{s}{Mglssymbol}%
14993 {\@glssymbolorGls}{\@glssymbolorgls}{\@glssymbolorGls}{\@glssymbolorgls}%
```

\MGlssymbol

`\MGlssymbol[<options>]{<label>}[<insert>]`

As above but initial each element if it's not a symbol.

```
14994 \glstr@newmgl{s}{MGlssymbol}%
14995 {\@glssymbolorGls}{\@glssymbolorGls}{\@glssymbolorGls}{\@glssymbolorGls}%
```

\mglsfield

```
14996 \newcommand{\mglsfield}{\useri}
```

\@glsfieIdorgls

```
14997 \def\@glsfieIdorgls#1#2[#3]{%
14998   \glstrifhasfield{\mglsfield}{#2}%
14999   {\@glsdisp[#1]{#2}{\glscurrentfieldvalue#3}}%
15000   {\@gl{s@{#1}{#2}[#3]}}%
15001 }
```

\@GlsfieIdorgls

```
15002 \def\@GlsfieIdorgls#1#2[#3]{%
15003   \glstrifhasfield{\mglsfield}{#2}%
15004   {\@glsdisp[#1]{#2}{\xmakefirstuc\glscurrentfieldvalue#3}}%
15005   {\@Gls@{#1}{#2}[#3]}}%
15006 }
```

\mglsusefield

`\mglsusefield[<options>]{<label>}[<insert>]`

Use the field given by \useri.

```
15007 \glstr@newmgl{s}{mglsusefield}%
15008 {\@glsfieIdorgls}{\@glsfieIdorgls}{\@glsfieIdorgls}{\@glsfieIdorgls}%
```

\Mgl{s}usefield

`\Mgl{s}usefield[<options>]{<label>}[<insert>]`

As above but use initial cap for first element only.

```
15009 \glstr@newmgl{s}{Mgl{s}usefield}%
15010 {\@GlsfieIdorgls}{\@glsfieIdorgls}{\@GlsfieIdorgls}{\@glsfieIdorgls}%
```

\MGl{s}usefield

`\MGlseusefield[<options>]{<label>}[<insert>]`

As above but use initial cap for all elements.

```
15011 \glstr@newmgls{MGlseusefield}%
15012 {\@Glsfielddorgls}{\@Glsfielddorgls}{\@Glsfielddorgls}{\@Glsfielddorgls}%
```

Use commands provided by glossaries-prefix if it has been loaded.

`\mpglsWarning`

```
15013 \newcommand*{\mpglsWarning}{%
15014   \GlossariesExtraWarning{glossaries-prefix.sty is required for
15015   \string\mpgls\space family of commands (either load after
15016   glossaries-extra.sty or use the ‘prefix’ package option)}%
15017 }
```

`\@pglsorgls`

```
15018 \def\@pglsorgls#1#2[#3]{%
15019   \ifdef\@pgls@{\@pgls@{#1}{#2}[#3]}\mpglsWarning\@gls@{#1}{#2}[#3]}%
15020 }
```

`\@pglsorglspl`

```
15021 \def\@pglsorglspl#1#2[#3]{%
15022   \ifdef\@pglspl@{\@pglspl@{#1}{#2}[#3]}\mpglsWarning\@glspl@{#1}{#2}[#3]}%
15023 }
```

`\@Pglsorgls`

```
15024 \def\@Pglsorgls#1#2[#3]{%
15025   \ifdef\@Pgls@{\@Pgls@{#1}{#2}[#3]}\mpglsWarning\@Gls@{#1}{#2}[#3]}%
15026 }
```

`\@pglsorglspl`

```
15027 \def\@pglsorglspl#1#2[#3]{%
15028   \ifdef\@pglspl@{\@pglspl@{#1}{#2}[#3]}\mpglsWarning\@glspl@{#1}{#2}[#3]}%
15029 }
```

`\@Pglsorglspl`

```
15030 \def\@Pglsorglspl#1#2[#3]{%
15031   \ifdef\@Pglspl@{\@Pglspl@{#1}{#2}[#3]}\mpglsWarning\@Glspl@{#1}{#2}[#3]}%
15032 }
```

`\@PGLSorgls`

```
15033 \def\@PGLSorgls#1#2[#3]{%
15034   \ifdef\@PGLS@{\@PGLS@{#1}{#2}[#3]}\mpglsWarning\@GLS@{#1}{#2}[#3]}%
15035 }
```

`\@PGLSorglspl`

```
15036 \def\@PGLSorglspl#1#2[#3]{%
15037   \ifdef\@PGLSpl@{\@PGLSpl@{#1}{#2}[#3]}\mpglsWarning\@GLSpl@{#1}{#2}[#3]}%
15038 }
```

`\mpgls`

`\mpgls[<options>]{<label>}[<insert>]`

Use `\pgls` for the first element and `\gls` for the remainder.

15039 `\glsxtr@newmgls{mpgls}{\@pglsorgls@}{\@gls@}{\@pglsorgls@}{\@gls@}%`

`\mpglspl`

`\mpglspl[<options>]{<label>}[<insert>]`

Use `\pglspl` for the first element and `\glspl` for the remainder.

15040 `\glsxtr@newmgls{mpglspl}{\@pglsorglspl@}{\@glspl@}{\@pglsorglspl@}{\@glspl@}%`

`\mpglsmainpl`

`\mpglsmainpl[<options>]{<label>}[<insert>]`

Only use plural for main element and only use prefixing command for first element.

15041 `\glsxtr@newmgls{mpglsmainpl}{\@pglsorgls@}{\@gls@}{\@pglsorglspl@}{\@glspl@}%`

`\Mpgls`

`\Mpgls[<options>]{<label>}[<insert>]`

Use `\Pgls` for the first element and `\gls` for the remainder.

15042 `\glsxtr@newmgls{Mpgls}{\@Pglsorgls@}{\@gls@}{\@Pglsorgls@}{\@gls@}%`

`\Mpglspl`

`\Mpglspl[<options>]{<label>}[<insert>]`

Use `\Pglspl` for the first element and `\glspl` for the remainder.

15043 `\glsxtr@newmgls{Mpglspl}{\@Pglsorglspl@}{\@glspl@}{\@Pglsorglspl@}{\@glspl@}%`

`\Mpglsmainpl`

`\Mpglsmainpl[<options>]{<label>}[<insert>]`

Only use plural for main element and only use first letter uppercase prefixing command for first element.

15044 `\glsxtr@newmgls{Mpglsmainpl}{\@Pglsorgls@}{\@gls@}{\@Pglsorglspl@}{\@glspl@}%`

\MPGls

\MPGls[*<options>*]{*<label>*}[*<insert>*]

Use \PglS for the first element and \Gls for the remainder.

15045 \glxtr@newmgls{MPGls}{\@Pglsorgls@}{\@Gls@}{\@Pglsorgls@}{\@Gls@}%

\MPGlspl

\MPGlspl[*<options>*]{*<label>*}[*<insert>*]

Use \PglSpl for the first element and \Glspl for the remainder.

15046 \glxtr@newmgls{MPGlspl}{\@Pglsorglspl@}{\@Glspl@}{\@Pglsorglspl@}{\@Glspl@}%

\MPGlsmainpl

\MPGlsmainpl[*<options>*]{*<label>*}[*<insert>*]

Only use plural for main element and first letter uppercase all elements.

15047 \glxtr@newmgls{MPGlsmainpl}{\@Pglsorgls@}{\@Gls@}{\@Pglsorglspl@}{\@Glspl@}%

\MPGLS

\MPGLS[*<options>*]{*<label>*}[*<insert>*]

Use \PGLS for the first element and \GLS for the remainder.

15048 \glxtr@newmgls{MPGLS}{\@PGLSorgls@}{\@GLS@}{\@PGLSorgls@}{\@GLS@}%

\MPGLSpl

\MPGLSpl[*<options>*]{*<label>*}[*<insert>*]

Use \PGLSpl for the first element and \GLSpl for the remainder.

15049 \glxtr@newmgls{MPGLSpl}{\@PGLSorglspl@}{\@GLSpl@}{\@PGLSorglspl@}{\@GLSpl@}%

\MPGLSmainpl

\MPGLSmainpl[*<options>*]{*<label>*}[*<insert>*]

Only use plural for main element and uppercase all elements.

15050 \glxtr@newmgls{MPGLSmainpl}{\@PGLSorgls@}{\@GLS@}{\@PGLSorglspl@}{\@GLSpl@}%

Not currently implementing any other variations.

1.10 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

sariesExtraLang

```
15051 \newcommand*{\RequireGlossariesExtraLang}[1]{%
15052   \ifundefined{ver@glossariesxtr-#1.1df}{\input{glossariesxtr-#1.1df}}{}}%
15053 }
```

sariesExtraLang

```
15054 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
15055   \ProvidesFile{glossariesxtr-#1.1df}%
15056 }
```

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. However, with `bib2gls` it might be useful to provide custom rules for a particular locale.)

xtr@loaddialect The dialect label should be stored in `\this@dialect` before using this command.

```
15057 \newcommand{\glxtr@loaddialect}{%
15058   \IfTrackedLanguageFileExists{\this@dialect}%
15059   {glossariesxtr-}% prefix
15060   {.1df}%
15061   {%
15062     \RequireGlossariesExtraLang{\CurrentTrackedTag}%
15063   }%
15064   {}% not found
```

If `glossaries-extra-bib2gls` has been loaded, `\glxtr@dialecthook` will check for the associated script, otherwise it will do nothing.

```
15065   \glxtr@dialecthook
15066 }

15067 \@ifpackageloaded{tracklang} {%
15068   \AnyTrackedLanguages
15069   {%
15070     \ForEachTrackedDialect{\this@dialect}{\glxtr@loaddialect}%
15071   }%
15072   {}%
15073 } {}
```

Load `glossaries-extra-stylemods` if required.

```
15074 \@glxtr@redefstyles
    and set the style:
15075 \@glxtr@do@style
```

1.11 glossaries-extra-bib2gls.sty

This package provides additional support for bib2gls and is automatically loaded by the record option.

```
15076 \NeedsTeXFormat{LaTeX2e}
15077 \ProvidesPackage{glossaries-extra-bib2gls}[2021/11/22 v1.48 (NLCT)]
```

Provide convenient shortcut commands for predefined glossary types.

ntunsrtaacronyms

```
15078 \ifglssaacronym
15079   \providecommand*\printunsrtaacronyms[1][]{%
15080     \printunsrtglossary[type=\acronymtype,#1]}%
15081 \fi
```

printunsrtindex

```
15082 \ifglossaryexists{index}
15083 {
15084   \providecommand*\printunsrtindex[1][]{%
15085     \printunsrtglossary[type=index,#1]}%
15086 }{}
```

intunsrtsymbols

```
15087 \ifglossaryexists{symbols}
15088 {
15089   \providecommand*\printunsrtsymbols[1][]{%
15090     \printunsrtglossary[type=symbols,#1]}%
15091 }{}
```

intunsrtnumbers

```
15092 \ifglossaryexists{numbers}
15093 {
15094   \providecommand*\printunsrtnumbers[1][]{%
15095     \printunsrtglossary[type=numbers,#1]}%
15096 }{}
```

rtaabbreviations

```
15097 \ifglossaryexists{abbreviations}
15098 {
15099   \providecommand*\printunsrtaabbreviations[1][]{%
15100     \printunsrtglossary[type=abbreviations,#1]}%
15101 }{}
```

splaynumberlist Allow \glsdisplaynumberlist and make it robust.

```
15102 \renewcommand*\glsdisplaynumberlist[1]{%
15103   \glsdoifexists{#1}%
15104   {%
15105     {\let\bibglssdelimN\glsnumlistsep
15106      \let\bibglslastDelimN\glsnumlistlastsep
```

```

15107      \glxtrusefield{#1}{location}%
15108      }%
15109      }%
15110 }
15111 \robustify\glsdisplaynumberlist

```

entrynumberlist

```

15112 \renewcommand*{\glsentrynumberlist}[1]{\glxtrusefield{#1}{location}}

```

These are some convenient macros for use with custom rules.

\glshex

```

15113 \newcommand*{\glshex}{\string\u}

```

lscapturedgroup

```

15114 \newcommand*{\glscapturedgroup}{\string\$}

```

nZeroChildCount For use with bib2gls's save-child-count resource option.

```

15115 \newcommand*{\GlsXtrIfHasNonZeroChildCount}{%
15116   \@ifstar\s@GlsXtrIfHasNonZeroChildCount\@GlsXtrIfHasNonZeroChildCount
15117 }

```

nZeroChildCount

```

15118 \newcommand*{\@GlsXtrIfHasNonZeroChildCount}[3]{%
15119   \@GlsXtrIfFieldNonZero{childcount}{#1}{#2}{#3}%
15120 }

```

nZeroChildCount

```

15121 \newcommand*{\s@GlsXtrIfHasNonZeroChildCount}[3]{%
15122   \s@GlsXtrIfFieldNonZero{childcount}{#1}{#2}{#3}%
15123 }

```

rprovidecommand For use in @preamble, this behaves like \providecommand in the document but like \renewcommand in bib2gls.

```

15124 \newcommand*{\glxtrprovidecommand}{\providecommand}

```

glsrenewcommand Like \renewcommand but only generates a warning rather than an error if the command isn't defined.

```

15125 \newcommand*{\glsrenewcommand}{\@star@or@long\glxtr@renewcommand}

```

tr@renewcommand

```

15126 \newcommand*{\glxtr@renewcommand}[1]{%
15127   \begingroup \escapechar\m@ne\xdef\@gtempa{\string#1}\endgroup
15128   \expandafter\@ifundefined\@gtempa
15129   {%
15130     \GlossariesExtraWarning{can't redefine \noexpand#1(not already defined)}%
15131   }%
15132   \relax

```



```

15133 \relax
15134 \let\@ifdefinable\@rc@ifdefinable
15135 \newcommand#1%
15136 }

```

lossarylocation For use with indexcounter and bib2gls.

```
15137 \newcommand*\GlsXtr@wrglossarylocation}[2]{#1}
```

indexCounterLink

`\GlsXtrIndexCounterLink{<text>}{<label>}`

For use with indexcounter and bib2gls.

```

15138 \ifdef\hyperref
15139 {%
15140   \newcommand*\GlsXtrIndexCounterLink}[2]{%
15141     \glxtrifhasfield{indexcounter}{#2}%
15142     {\hyperref[wrglossary.\glscurrentfieldvalue]{#1}}%
15143     {#1}%
15144   }
15145 }
15146 {
15147   \newcommand*\GlsXtrIndexCounterLink}[2]{#1}
15148 }

```

GlsXtrDualField

`\GlsXtrDualField`

The internal field used to store the dual label. The dual-field defaults to dual if no value is supplied so that's used as the default.

```
15149 \newcommand*\GlsXtrDualField}{dual}
```

XtrDualBackLink

`\GlsXtrDualBackLink{<text>}{<label>}`

Adds a hyperlink to the dual entry.

```

15150 \newcommand*\GlsXtrDualBackLink}[2]{%
15151   \glxtrifhasfield{\GlsXtrDualField}{#2}%
15152   {\glshyperlink[#1]{\glscurrentfieldvalue}}%
15153   {#2}%
15154 }

```

TeXEntryAliases Convenient shortcut for use with entry-type-aliases to alias standard BibTeX entry types to @bibtexentry.

```

15155 \newcommand*{\GlsXtrBibTeXEntryAliases}{%
15156   article=bibtexentry,
15157   book=bibtexentry,
15158   booklet=bibtexentry,
15159   conference=bibtexentry,
15160   inbook=bibtexentry,
15161   incollection=bibtexentry,
15162   inproceedings=bibtexentry,
15163   manual=bibtexentry,
15164   mastersthesis=bibtexentry,
15165   misc=bibtexentry,
15166   phdthesis=bibtexentry,
15167   proceedings=bibtexentry,
15168   techreport=bibtexentry,
15169   unpublished=bibtexentry
15170 }
```

ideBibTeXFields Convenient shortcut to define the standard BibTeX fields.

```

15171 \newcommand*{\GlsXtrProvideBibTeXFields}{%
15172   \glsaddstoragekey{address}{}{\glsxtrbibaddress}%
15173   \glsaddstoragekey{author}{}{\glsxtrbibauthor}%
15174   \glsaddstoragekey{booktitle}{}{\glsxtrbibbooktitle}%
15175   \glsaddstoragekey{chapter}{}{\glsxtrbibchapter}%
15176   \glsaddstoragekey{edition}{}{\glsxtrbibedition}%
15177   \glsaddstoragekey{howpublished}{}{\glsxtrbibhowpublished}%
15178   \glsaddstoragekey{institution}{}{\glsxtrbibinstitution}%
15179   \glsaddstoragekey{journal}{}{\glsxtrbibjournal}%
15180   \glsaddstoragekey{month}{}{\glsxtrbibmonth}%
15181   \glsaddstoragekey{note}{}{\glsxtrbibnote}%
15182   \glsaddstoragekey{number}{}{\glsxtrbibnumber}%
15183   \glsaddstoragekey{organization}{}{\glsxtrbiborganization}%
15184   \glsaddstoragekey{pages}{}{\glsxtrbibpages}%
15185   \glsaddstoragekey{publisher}{}{\glsxtrbibpublisher}%
15186   \glsaddstoragekey{school}{}{\glsxtrbibschooll}%
15187   \glsaddstoragekey{series}{}{\glsxtrbibseries}%
15188   \glsaddstoragekey{title}{}{\glsxtrbibtitle}%
15189   \glsaddstoragekey{bibtextype}{}{\glsxtrbibtype}%
15190   \glsaddstoragekey{volume}{}{\glsxtrbibvolume}%
15191 }
```

Multiple supplementary references are only supported with bib2gls.

ltisupplocation This is like \glsxtrsupphypernumber but the second argument is the external file name (which isn't obtained from the externallocation attribute). The third argument is the formatting (encap) control sequence *name*. This is ignored by default, but is set by bib2gls to the original encap in case it's required.

```

15192 \newcommand*{\glsxtrmultisupplocation}[3]{%
```

```

15193 {%
15194   \def\glstrsupplocationurl{#2}%
15195   \glshypernumber{#1}%
15196 }%
15197 }

```

rdisplaysuppl

$\backslash\text{glstrdisplaysuppl}\langle\text{prefix}\rangle\langle\text{counter}\rangle\langle\text{format}\rangle\langle\text{src}\rangle\langle\text{location}\rangle$

This is like $\backslash\text{glsnoidxdisplayloc}$ but is used for supplementary locations and so requires an extra argument.

```

15198 \newcommand*\glstrdisplaysuppl[5]{%
15199   \setentrycounter{#1}{#2}%
15200   \glstrmultisupplocation{#5}{#4}{#3}%
15201 }

```

$\backslash\text{glstrdisplaylocnameref}$ $\backslash\text{glstrdisplaylocnameref}\langle\text{prefix}\rangle\langle\text{counter}\rangle\langle\text{format}\rangle\langle\text{location}\rangle\langle\text{name}\rangle\langle\text{href}\rangle\langle\text{hcounter}\rangle\langle\text{external file}\rangle$ Used with the [nameref] record package option. The $\langle\text{href}\rangle$ argument was obtained from $\backslash\text{@currentHref}$ and the $\langle\text{hcounter}\rangle$ argument was obtained from $\backslash\text{theHentrycounter}$, which is more reliable. If hyperref hasn't been loaded, this just behaves like $\backslash\text{glsnoidxdisplayloc}$.

```

15202 \ifundef\hyperlink
15203 {
15204   \newcommand*\glstrdisplaylocnameref[8]{%
15205     \glsnoidxdisplayloc{#1}{#2}{#3}{#4}%
15206   }
15207 }
15208 {

```

Default action uses $\langle\text{hcounter}\rangle$. Equations and pages typically don't have a title, so check the counter name (otherwise the title may section or chapter title, which may be confusing). As from v1.42, this now checks if the control sequence $\backslash\text{glstr}\langle\text{counter}\rangle\text{locfmt}$ is defined.

```

15209   \newcommand*\glstrdisplaylocnameref[8]{%
15210     \ifcsdef{glstr#2locfmt}%
15211       {\glstrnameref\link{#3}{#4}{#2.#7}{#8}}%
15212     {%
15213       \ifstrempy{#5}%
15214       {%

```

No title, so just use the location as the link text.

```

15215       \glstrnameref\link{#3}{#4}{#2.#7}{#8}%
15216     }%
15217   {%
15218     \ifstrequal{#2}{page}%
15219     {\glstrnameref\link{#3}{#4}{#2.#7}{#8}}%
15220     {\glstrnameref\link{#3}{#5}{#2.#7}{#8}}%

```

```

15221     }%
15222     }%
15223   }
15224 }

```

requisitionlocfmt

```
\glxsxtrequisitionlocfmt{<location>}{<title>}
```

```
15225 \newcommand*{\glxsxtrequisitionlocfmt}[2]{( #1)}
```

sxtrnamerefink

```
\glsxtrfmtnamerefink{<format>}{<title>}{<href>}{<external file>}
```

```
15226 \newcommand*{\glsxtrnamerefink}[4]{%
```

Locally change \glshypernumber to \@firstofone to remove the normal location hyperlink.

```
15227   \begingroup
```

```
15228     \let\glshypernumber\@firstofone
```

If the <external file> argument is empty, an internal link is used, otherwise an external one is needed.

```
15229     \ifstrempy{#4}%
```

```
15230       {\glsxtrfmtinternalnameref{#3}{#1}{#2}}%
```

```
15231       {\glsxtrfmtexternalnameref{#3}{#1}{#2}{#4}}%
```

```
15232   \endgroup
```

```
15233 }
```

sxtrnameloclink

```
\glsxtrnamerefloclink{<prefix>}{<counter>}{<format>}{<location>}{<text>}{<external file>}
```

Like \@gls@numberlink, this creates a hyperlink to the target obtained from the prefix, counter and location but uses <text> as the hyperlink text. As with regular indexing, this will fail if the target name can't be formed by prefixing the location value.

```
15234 \newcommand{\glsxtrnameloclink}[6]{%
```

```
15235   \begingroup
```

```
15236     \setentrycounter[#1]{#2}%
```

```
15237     \def\glxtr@locationhypertext{#5}%
```

```
15238     \let\glshypernumber\@firstofone
```

```
15239     \def\@glsnumberformat{#3}%
```

```
15240     \def\glxtrsupplocationurl{#6}%
```

```

15241 \toks@={}%
15242 \@glsxtr@bibgls@removespaces#4 \@nil
15243 \endgroup
15244 }

```

ls@removespaces

```

15245 \def\@glsxtr@bibgls@removespaces#1 #2\@nil{%
15246 \toks@=\expandafter{\the\toks@#1}%
15247 \ifx\#2\%

15248 \edef\@glo@tmp{\the\toks@}%
15249 \ifx\@glo@tmp\empty
15250 \else
15251 \protected@edef\@glo@tmp{\glstrycounter\@glo@counterprefix\the\toks@}%
15252 \ifdefined\glsxtrsupplocationurl
15253 {%
15254 \expandafter\glsxtrfmtinternalnameref\expandafter{\@glo@tmp}%
15255 {\@glsnumberformat}\glsxtr@locationhypertext}%
15256 }%
15257 {%
15258 \expandafter\glsxtrfmtexternalnameref\expandafter{\@glo@tmp}%
15259 {\@glsnumberformat}\glsxtr@locationhypertext}\glsxtrsupplocationurl}%
15260 }%
15261 \fi
15262 \else
15263 \@gls@ReturnAfterFi{%
15264 \@glsxtr@bibgls@removespaces#2\@nil
15265 }%
15266 \fi
15267 }

```

internalnameref

```
\glsxtrfmtinternalnameloc{<target>}{<format>}{<title>}
```

```

15268 \newcommand*\glsxtrfmtinternalnameref}[3]{%
15269 \csuse{#2}{\glsdohyperlink{#1}{#3}}%
15270 }

```

externalnameref

```
\glsxtrfmtexternalnameloc{<target>}{<format>}{<title>}{<file>}
```

```

15271 \newcommand*\glsxtrfmtexternalnameref}[4]{%
15272 \csuse{#2}{\hyperref{#4}{#1}{#3}}%
15273 }

```

glsxtrSetWidest

`\glsxtrSetWidest{<type>}{<level>}{<text>}`

As from **bib2gls** v1.8, this is used by the `set-widest` resource option for the `alttree` and the styles provided by the `glossary-longextra` package.

```
15274 \newcommand*{\glsxtrSetWidest}[3]{%
```

Check which style options have been provided. (The style packages may not have been loaded.)

```
15275 \ifdef\glsupdatewidest
15276 {%
15277 \ifdef\glslongextraUpdateWidest
15278 {%
```

Relevant style packages all loaded. If the *<type>* has been given, append to glossary preamble.

```
15279 \ifstrempy{#1}
15280 {%
15281 \glsupdatewidest[#2]{#3}%
15282 \ifnum#2=0\relax
15283 \glslongextraUpdateWidest{#3}%
15284 \else
15285 \glslongextraUpdateWidestChild{#2}{#3}%
15286 \fi
15287 }%
15288 {%
15289 \apptoglossarypreamble[#1]{\glsupdatewidest[#2]{#3}}%
15290 \ifnum#2=0\relax
15291 \apptoglossarypreamble[#1]{\glslongextraUpdateWidest{#3}}%
15292 \else
15293 \apptoglossarypreamble[#1]{\glslongextraUpdateWidestChild{#2}{#3}}%
15294 \fi
15295 }%
15296 }%
15297 {%
```

Only `alttree`.

```
15298 \ifstrempy{#1}
15299 {%
15300 \glsupdatewidest[#2]{#3}%
15301 }%
15302 {%
15303 \apptoglossarypreamble[#1]{\glsupdatewidest[#2]{#3}}%
15304 }%
15305 }%
15306 }%
15307 {%
```

`\glsupdatewidest` hasn't been defined. This could just mean that the `glossaries-extra-stylemods` package hasn't been loaded.

```

15308 \ifdef\glsetwidest
15309 {%
15310 \ifdef\glslongextraUpdateWidest
15311 {%

```

Relevant glossary-tree and glossary-longextra have been loaded. If the *<type>* has been given, append to glossary preamble.

```

15312 \ifstrempy{#1}
15313 {%
15314 \glsetwidest[#2]{#3}%
15315 \ifnum#2=0\relax
15316 \glslongextraUpdateWidest{#3}%
15317 \else
15318 \glslongextraUpdateWidestChild{#2}{#3}%
15319 \fi
15320 }%
15321 {%
15322 \apptoglossarypreamble[#1]{\glsetwidest[#2]{#3}}%
15323 \ifnum#2=0\relax
15324 \apptoglossarypreamble[#1]{\glslongextraUpdateWidest{#3}}%
15325 \else
15326 \apptoglossarypreamble[#1]{\glslongextraUpdateWidestChild{#2}{#3}}%
15327 \fi
15328 }%
15329 }%
15330 {%

```

Only altree.

```

15331 \ifstrempy{#1}
15332 {%
15333 \glsetwidest[#2]{#3}%
15334 }%
15335 {%
15336 \apptoglossarypreamble[#1]{\glsetwidest[#2]{#3}}%
15337 }%
15338 }%
15339 }%
15340 {%
15341 \ifdef\glslongextraUpdateWidest
15342 {%

```

glossary-longextra has been loaded.

```

15343 \ifstrempy{#1}
15344 {%
15345 \ifnum#2=0\relax
15346 \glslongextraUpdateWidest{#3}%
15347 \else
15348 \glslongextraUpdateWidestChild{#2}{#3}%
15349 \fi
15350 }%
15351 {%

```

```

15352         \ifnum#2=0\relax
15353             \apptoglossarypreamble[#1]{\glslongextraUpdateWidest{#3}}%
15354         \else
15355             \apptoglossarypreamble[#1]{\glslongextraUpdateWidestChild{#2}{#3}}%
15356         \fi
15357     }%
15358 }%

Neither glossary-tree nor glossary-longextra have been loaded. Do nothing.

15359     {}%
15360 }%
15361 }%
15362 }

```

tWidestFallback

`\glsxtrSetWidestFallback{<max depth>}{<list>}`

Used when **bib2gls** can't determine the widest name. The *<list>* argument is a comma-separated list of glossary labels. The *<max depth>* refers to the maximum hierarchical depth. This will either be 0 (only top-level entries) or 2 (up to two child-levels).

```

15363 \newcommand*{\glsxtrSetWidestFallback}[2]{%
15364     \ifnum#1=0\relax
15365         \ifdef\glsFindWidestTopLevelName
15366             {%
15367                 \glsFindWidestTopLevelName[#2]%
15368             }%
15369             {%
15370                 \GlossariesExtraWarning{You need stylemods={tree} to
15371                     provide a fallback for set-widest}%
15372             }%
15373         \else
15374             \ifdef\glsFindWidestLevelTwo
15375                 {%
15376                     \glsFindWidestLevelTwo[#2]%
15377                     \ifdef\glslongextraUpdateWidestChild
15378                         {%
15379                             \glslongextraUpdateWidestChild{#1}{\csuse{@glswidestnamei}}%
15380                             \glslongextraUpdateWidestChild{#1}{\csuse{@glswidestnameii}}%
15381                         }%
15382                     {}%
15383                 }%
15384                 {%
15385                     \GlossariesExtraWarning{You need stylemods={tree} to
15386                         provide a fallback for set-widest}%
15387                 }%
15388             \fi
15389 }

```


`r@labelprefixes` List of label prefixes.

```
15390 \newcommand*{\@glxstr@labelprefixes}{}
```

`arlabelprefixes` List of label prefixes.

```
15391 \newcommand*{\glxstrclearlabelprefixes}{%
15392   \renewcommand*{\@glxstr@labelprefixes}{}}%
15393 }
```

`raddlabelprefix` Add prefix to the list. These should be added in the order of precedence with the last one as a fallback. This doesn't check against duplicates as it may be useful to replicate a prefix at the end as the fallback.

```
15394 \newcommand*{\glxstraddlabelprefix}[1]{%
15395   \ifstrempy{#1}%
15396   {\glxstraddlabelprefix{\empty}}%
15397   {%
15398     \ifdefempty\@glxstr@labelprefixes
15399     {\def\@glxstr@labelprefixes{#1}}%
15400     {\appto\@glxstr@labelprefixes{,#1}}%
15401   }%
15402 }
```

`pendlabelprefix` Inserts at the start of the list.

```
15403 \newcommand*{\glxstrprependlabelprefix}[1]{%
15404   \ifstrempy{#1}%
15405   {\glxstrprependlabelprefix{\empty}}%
15406   {%
15407     \ifdefempty\@glxstr@labelprefixes
15408     {\def\@glxstr@labelprefixes{#1}}%
15409     {\preto\@glxstr@labelprefixes{#1,}}%
15410   }%
15411 }
```

`labelprefixlist`

`\glxstrifinlabelprefixlist{<prefix>}{<true>}{<false>}`

Test if the given prefix is in the list.

```
15412 \newcommand*{\glxstrifinlabelprefixlist}[3]{%
15413   \ifstrempy{#1}%
15414   {\glxstrifinlabelprefixlist{\empty}{#2}{#3}}%
15415   {%
15416     \DTLifinlist{#1}{\@glxstr@labelprefixes}{#2}{#3}%
15417   }%
15418 }
```

prefixlabellist This is provided for the benefit of **bib2gls**. It's possible that the user may add more prefixes after the start of the document, but that can lead to inconsistencies. The final element of the list (the fallback) is the only prefix of interest for **bib2gls**.

```

15419 \AtBeginDocument{%
15420   \protected@write\@auxout{}\string\providecommand{\string\@glstr@prefixlabellist}[1]{}%
15421   \protected@write\@auxout{}\string\@glstr@prefixlabellist{\@glstr@labelprefixes}}%
15422 }

```

t@prefixedlabel Iterate through all the prefixes and find the first prefix and label combination that exists. If none found, this could mean that it's the first \TeX run, so the last prefix in the list needs to be the fallback one. Grouping is used in case of a nested for loop.

```

15423 \newcommand*{\@glstr@get@prefixedlabel}[1]{%
15424   \begingroup

       Initialise to the unprefixed label in the event that the list is empty.
15425   \protected@edef\@gls@thislabel{#1}%
15426   \@for\@glstr@prefix:=\@glstr@labelprefixes\do
15427   {%
15428     \protected@edef\@gls@thislabel{\@glstr@prefix#1}%
15429     \ifglstryexists{\@gls@thislabel}\@endfortrue{}}%
15430   }%

15431   \edef\@glo@tmp{\endgroup\noexpand\def\noexpand\@gls@thislabel{\@gls@thislabel}}\@glo@tmp
15432 }

```

\dglS Like \gls but tries the prefixes. (Can't use \pgls as that's provided by glossaries-prefix.) Since this command is designed for bib2gls's dual entry system, the "d" stands for "dual".

```

15433 \newrobustcmd*{\dglS}{\@gls@hyp@opt\dglS}

```

\@dglS

```

15434 \newcommand*{\@dglS}[2][{}]{%
15435   \@glstr@get@prefixedlabel{#2}%
15436   \new@ifnextchar[{\@gls@{#1}\@gls@thislabel}]{\@gls@{#1}\@gls@thislabel}[{}]}%
15437 }

```

\dglSpl

```

15438 \newrobustcmd*{\dglSpl}{\@gls@hyp@opt\dglSpl}

```

\@dglSpl

```

15439 \newcommand*{\@dglSpl}[2][{}]{%
15440   \@glstr@get@prefixedlabel{#2}%
15441   \new@ifnextchar[{\@glspl@{#1}\@gls@thislabel}]{\@glspl@{#1}\@gls@thislabel}[{}]}%
15442 }

```

\dGls

```

15443 \newrobustcmd*{\dGls}{\@gls@hyp@opt\dGls}

```

\@dGls

```
15444 \newcommand*{\@dGls}[2] [] {%
15445   \@glsxtr@get@prefixedlabel{#2}%
15446   \new@ifnextchar[{\@Gls@{#1}{\@gls@thislabel}}{\@Gls@{#1}{\@gls@thislabel}} []}%
15447 }
```

\dGlspl

```
15448 \newrobustcmd*{\dGlspl}{\@gls@hyp@opt\dGlspl}
```

\@dGlspl

```
15449 \newcommand*{\@dGlspl}[2] [] {%
15450   \@glsxtr@get@prefixedlabel{#2}%
15451   \new@ifnextchar[{\@Glspl@{#1}{\@gls@thislabel}}{\@Glspl@{#1}{\@gls@thislabel}} []}%
15452 }
```

\dGLS

```
15453 \newrobustcmd*{\dGLS}{\@gls@hyp@opt\dGLS}
```

\@dGLS

```
15454 \newcommand*{\@dGLS}[2] [] {%
15455   \@glsxtr@get@prefixedlabel{#2}%
15456   \new@ifnextchar[{\@GLS@{#1}{\@gls@thislabel}}{\@GLS@{#1}{\@gls@thislabel}} []}%
15457 }
```

\dGLSpl

```
15458 \newrobustcmd*{\dGLSpl}{\@gls@hyp@opt\dGLSpl}
```

\@dGLSpl

```
15459 \newcommand*{\@dGLSpl}[2] [] {%
15460   \@glsxtr@get@prefixedlabel{#2}%
15461   \new@ifnextchar[{\@GLSpl@{#1}{\@gls@thislabel}}{\@GLSpl@{#1}{\@gls@thislabel}} []}%
15462 }
```

\dglslink Like \glslink but tries the prefixes.

```
15463 \newrobustcmd*{\dglslink}[3] [] {%
15464   \@glsxtr@get@prefixedlabel{#2}%
15465   \glslink[#1]{\@gls@thislabel}{#3}%
15466 }
```

\dglsdisp Like \glsdisp but tries the prefixes.

```
15467 \newrobustcmd*{\dglsdisp}[3] [] {%
15468   \@glsxtr@get@prefixedlabel{#2}%
15469   \glsdisp[#1]{\@gls@thislabel}{#3}%
15470 }
```

Multi (compound/combined) entry commands used by bib2gls.

tryadjustedname

`\glstrmultientryadjustedname{<list1>}{<name>}{<list2>}{<label>}`

This command is used by bib2gls when it adjusts the name field of an entry that's been identified as a main entry in the multi-entry set *<label>*.

The final argument *<label>* is the multi-entry label from which the set was obtained. The first argument *<list1>* is the list of other labels that come before the main label. The third argument *<list2>* is the remaining list of other labels. The *<name>* argument is the previous name before adjustment.

```
15471 \newrobustcmd*{\glstrmultientryadjustedname}[4]{%
15472   \bgroup
15473   \let\@glstrmultientryadjustednamesep\glstrmultientryadjustednamesep
15474   \let\@glstrmultientryadjustednamepresep\glstrmultientryadjustednamepresep
15475   \let\@glstrmultientryadjustednamepostsep\glstrmultientryadjustednamepostsep
15476   \let\@glstrmultientryadjustednameother\glstrmultientryadjustednameother
15477   \let\@glstrmultientryadjustednamefmt\glstrmultientryadjustednamefmt
15478   \let\@glstrmultientryadjustednamefirstother\glstrmultientryadjustednameother
15479   \let\@glstrmultientryadjustednamefirstfmt\glstrmultientryadjustednamefmt
15480   \@glstrmultientryadjustedname{#1}{#2}{#3}{#4}%
15481   \egroup
15482 }
```

tryadjustedname First letter upper case

```
15483 \newrobustcmd*{\Glsxtrmultientryadjustedname}[4]{%
15484   \bgroup
15485   \let\@glstrmultientryadjustednamesep\glstrmultientryadjustednamesep
15486   \let\@glstrmultientryadjustednamepresep\glstrmultientryadjustednamepresep
15487   \let\@glstrmultientryadjustednamepostsep\glstrmultientryadjustednamepostsep
15488   \let\@glstrmultientryadjustednameother\glstrmultientryadjustednameother
15489   \let\@glstrmultientryadjustednamefmt\glstrmultientryadjustednamefmt
15490   \let\@glstrmultientryadjustednamefirstother\Glsxtrmultientryadjustednameother
15491   \let\@glstrmultientryadjustednamefirstfmt\Glsxtrmultientryadjustednamefmt
15492   \@glstrmultientryadjustedname{#1}{#2}{#3}{#4}%
15493   \egroup
15494 }
```

tryadjustedname Title case

```
15495 \newrobustcmd*{\GlsXtrmultientryadjustedname}[4]{%
15496   \bgroup
15497   \let\@glstrmultientryadjustednamesep\glstrmultientryadjustednamesep
15498   \let\@glstrmultientryadjustednamepresep\glstrmultientryadjustednamepresep
15499   \let\@glstrmultientryadjustednamepostsep\glstrmultientryadjustednamepostsep
15500   \let\@glstrmultientryadjustednameother\GlsXtrmultientryadjustednameother
15501   \let\@glstrmultientryadjustednamefmt\GlsXtrmultientryadjustednamefmt
15502   \let\@glstrmultientryadjustednamefirstother\GlsXtrmultientryadjustednameother
15503   \let\@glstrmultientryadjustednamefirstfmt\GlsXtrmultientryadjustednamefmt
15504   \@glstrmultientryadjustedname{#1}{#2}{#3}{#4}%

```

```

15505 \egroup
15506 }

```

tryadjustedname All caps.

```

15507 \newrobustcmd*{\GLSxtrmultientryadjustedname}[4]{%
15508 \bgroup
15509 \let\@glsxtrmultientryadjustednamesep\glsxtrmultientryadjustednamesep
15510 \let\@glsxtrmultientryadjustednamepresep\glsxtrmultientryadjustednamepresep
15511 \let\@glsxtrmultientryadjustednamepostsep\glsxtrmultientryadjustednamepostsep
15512 \let\@glsxtrmultientryadjustednameother\GLSxtrmultientryadjustednameother
15513 \let\@glsxtrmultientryadjustednamefmt\GLSxtrmultientryadjustednamefmt
15514 \let\@glsxtrmultientryadjustednameother\GLSxtrmultientryadjustednameother
15515 \let\@glsxtrmultientryadjustednamefirstfmt\GLSxtrmultientryadjustednamefirstfmt
15516 \@glsxtrmultientryadjustedname{#1}{#2}{#3}{#4}%
15517 \egroup
15518 }

```

tryadjustedname

```

15519 \newcommand*{\@glsxtrmultientryadjustedname}[4]{%
15520 \letcs\mglscurrentmainlabel{\@glsc@combined@#4@main}%
15521 \letcs\mglscurrentmainlist{\@glsc@combined@#4@list}%
15522 \letcs\mglscurrentmainoptions{\@glsc@combined@#4@options}%
15523 \ifblank{#1}%
15524 {%
15525 \@glsxtrmultientryadjustednamefirstfmt{#2}%
15526 }%
15527 {%
15528 \def\@mgls@previouslabel{}%
15529 \let\@glsc@xtradjustedother\@glsxtrmultientryadjustednamefirstother
15530 \@for\mglscurrentlabel:=#1\do{%
15531 \ifx\@mgls@previouslabel\empty
15532 \else
15533 \@glsxtrmultientryadjustednamesep{\@mgls@previouslabel}{\mglscurrentlabel}%
15534 \fi
15535 \@glsc@xtradjustedother{\mglscurrentlabel}%
15536 \let\@mgls@previouslabel\mglscurrentlabel
15537 \let\@glsc@xtradjustedother\@glsxtrmultientryadjustednameother
15538 }%
15539 \@glsxtrmultientryadjustednamepresep{\@mgls@previouslabel}{\mglscurrentmainlabel}%
15540 \@glsxtrmultientryadjustednamefmt{#2}%
15541 }%
15542 \ifblank{#3}%
15543 {}%
15544 {%
15545 \let\@mgls@previouslabel\mglscurrentmainlabel
15546 \let\@glsc@xtrmultientryadjustednamesep\@glsxtrmultientryadjustednamepostsep
15547 \@for\mglscurrentlabel:=#3\do{%
15548 \@glsc@xtrmultientryadjustednamesep{\@mgls@previouslabel}{\mglscurrentlabel}%
15549 \@glsxtrmultientryadjustednameother{\mglscurrentlabel}%

```

```

15550      \let\@mgls@previouslabel\mglscurrentlabel
15551      \let\@gls@xtrmultientryadjustednamesep\@glsxtrmultientryadjustednamesep
15552      }%
15553      }%
15554 }

```

adjustednamesep

```
15555 \newcommand*{\glsxtrmultientryadjustednamesep}{\glscombinedfirstsepfirst}
```

ustednamepresep Separator before main name.

```
15556 \newcommand*{\glsxtrmultientryadjustednamepresep}{\glsxtrmultientryadjustednamesep}
```

stednamepostsep Separator after main name.

```
15557 \newcommand*{\glsxtrmultientryadjustednamepostsep}{\glsxtrmultientryadjustednamesep}
```

adjustednamefmt

```
15558 \newcommand*{\glsxtrmultientryadjustednamefmt}[1]{#1}
```

justednameother

```
15559 \newcommand*{\glsxtrmultientryadjustednameother}[1]{\glsentryname{#1}}
```

adjustednamefmt

```
15560 \newcommand*{\Glsxtrmultientryadjustednamefmt}[1]{\makefirstuc{#1}}
```

justednameother

```
15561 \newcommand*{\Glsxtrmultientryadjustednameother}[1]{\Glsentryname{#1}}
```

justednameother

```
15562 \newcommand*{\GlsXtrmultientryadjustednameother}[1]{%
15563   \glsentrytitlecase{#1}{name}}
```

adjustednamefmt

```

15564 \ifdef\glscapitalisewords
15565 {%
15566   \newcommand*{\GlsXtrmultientryadjustednamefmt}[1]{\glscapitalisewords{#1}}
15567 }
15568 {
15569   \newcommand*{\GlsXtrmultientryadjustednamefmt}[1]{\capitalisewords{#1}}
15570 }

```

justednameother

```

15571 \newcommand*{\GLSxtrmultientryadjustednameother}[1]{%
15572   \mfirstucMakeUppercase{\glsentryname{#1}}}

```

adjustednamefmt

```
15573 \newcommand*{\GLSxtrmultientryadjustednamefmt}[1]{\mfirstucMakeUppercase{#1}}
```

Provide missing Greek letters for use in maths mode. These commands are recognised by bib2gls and will be mapped to the Mathematical Greek Italic letters. This ensures that the Greek letters that have the same shape as Latin letters are kept with the other mathematical Greek letters for sorting purposes. The \LaTeX version of these commands (provided here) use an upright font for capitals and italic for lower case to provide a better match with the other Greek symbols provided by the kernel.

`\Alpha`

15574 `\providecommand*{\Alpha}{\mathrm{A}}`

`\Beta`

15575 `\providecommand*{\Beta}{\mathrm{B}}`

`\Epsilon`

15576 `\providecommand*{\Epsilon}{\mathrm{E}}`

`\Zeta`

15577 `\providecommand*{\Zeta}{\mathrm{Z}}`

`\Eta`

15578 `\providecommand*{\Eta}{\mathrm{H}}`

`\Iota`

15579 `\providecommand*{\Iota}{\mathrm{I}}`

`\Kappa`

15580 `\providecommand*{\Kappa}{\mathrm{K}}`

`\Mu`

15581 `\providecommand*{\Mu}{\mathrm{M}}`

`\Nu`

15582 `\providecommand*{\Nu}{\mathrm{N}}`

`\Omicron`

15583 `\providecommand*{\Omicron}{\mathrm{O}}`

`\Rho`

15584 `\providecommand*{\Rho}{\mathrm{P}}`

`\Tau`

15585 `\providecommand*{\Tau}{\mathrm{T}}`

`\Chi`

15586 `\providecommand*{\Chi}{\mathrm{X}}`

```

\Digamma
15587 \providecommand*\Digamma{\mathrm{F}}

\omicron
15588 \providecommand*\omicron{\mathit{o}}

        Provide corresponding upright characters if upgreek has been loaded. (The upper case
        characters are the same as above.)
15589 \@ifpackageloaded{upgreek}%
15590 {

\Upalpha
15591   \providecommand*\Upalpha{\mathrm{A}}

\Upbeta
15592   \providecommand*\Upbeta{\mathrm{B}}

\Upepsilon
15593   \providecommand*\Upepsilon{\mathrm{E}}

\Upzeta
15594   \providecommand*\Upzeta{\mathrm{Z}}

\Upeta
15595   \providecommand*\Upeta{\mathrm{H}}

\Upiota
15596   \providecommand*\Upiota{\mathrm{I}}

\Upkappa
15597   \providecommand*\Upkappa{\mathrm{K}}

\Upmu
15598   \providecommand*\Upmu{\mathrm{M}}

\Upnu
15599   \providecommand*\Upnu{\mathrm{N}}

\Upomicron
15600   \providecommand*\Upomicron{\mathrm{O}}

\Uprho
15601   \providecommand*\Uprho{\mathrm{P}}

\Uptau
15602   \providecommand*\Uptau{\mathrm{T}}

```


`\Upchi`

```
15603 \providecommand*\Upchi{\mathrm{X}}
```

`\upomicron`

```
15604 \providecommand*\upomicron{\mathrm{o}}
```

```
15605 }%
```

```
15606 {}% upgreek.sty not loaded
```

This package provides some basic rules, but it's not intended for complete coverage of all locales. The CLDR should provide the appropriate locale-sensitive rules. These macros are primarily to help construct custom rules to include, for example, Greek maths symbols mixed with Latin. For the full rule syntax, see the Java API for [RuleBaseCollator](#)

If you want to provide a rule-block for a particular locale to allow for customization within that locale, create a file called `glossariesxtr-⟨tag⟩.ldf` (where `⟨tag⟩` identifies the locale) and add similar commands. See the description of `\IfTrackedLanguageFileExists` in the `tracklang` manual for the allowed forms of `⟨tag⟩`. The simplest is to just use the root language label or ISO code. The file will then be automatically loaded by `glossaries-extra` if the document has support for that language.

When combining these blocks of rules, remember to separate them with the appropriate character. For example:

```
sort-rule={\glxtrcontrolrules
; \glxtrspacerules
; \glxtrnonprintablerules
; \glxtrcombiningdiacriticrules
, \glxtrhyphenrules
< \glxtrgeneralpuncrules
< \glxtrdigitrules
< \glxtrfractionrules
< \glxtrGeneralLatinIVrules
< \glxtrMathItalicGreekIrules
}
```

`xtrcontrolrules` These are control characters that are usually placed at the start of a rule in the ‘ignored characters’ section. These control characters are unlikely to appear in any entry fields but are provided for completeness. `\string` is used for punctuation characters in case they’ve been made active.

```
15607 \newcommand*\glxtrcontrolrules{%
15608 \string'\glshex 200B\string'\string=\glshex 200C\string=\glshex 200D
15609 \string=\glshex 200E\string=\glshex 200F\string=\glshex 0000\string=\glshex 0001
15610 \string=\glshex 0002\string=\glshex 0003\string=\glshex 0004\string=\glshex 0005
15611 \string=\glshex 0006\string=\glshex 0007\string=\glshex 0008
15612 \string=\string'\glshex 0009\string'\string=\string'\glshex 000B\string'
15613 \string=\glshex 000E\string=\glshex 000F\string=\string'\glshex
15614 0010\string'\string=\glshex 0011
15615 \string=\glshex 0012\string=\glshex 0013\string=\glshex 0014\string=\glshex 0015
15616 \string=\glshex 0016\string=\glshex 0017\string=\glshex 0018\string=\glshex 0019
```

```

15617 \string=\glshex 001A\string=\glshex 001B\string=\glshex 001C\string=\glshex 001D
15618 \string=\glshex 001E\string=\glshex 001F\string=\glshex 007F\string=\glshex 0080
15619 \string=\glshex 0081\string=\glshex 0082\string=\glshex 0083\string=\glshex 0084
15620 \string=\glshex 0085\string=\glshex 0086\string=\glshex 0087\string=\glshex 0088
15621 \string=\glshex 0089\string=\glshex 008A\string=\glshex 008B\string=\glshex 008C
15622 \string=\glshex 008D\string=\glshex 008E\string=\glshex 008F\string=\glshex 0090
15623 \string=\glshex 0091\string=\glshex 0092\string=\glshex 0093\string=\glshex 0094
15624 \string=\glshex 0095\string=\glshex 0096\string=\glshex 0097\string=\glshex 0098
15625 \string=\glshex 0099\string=\glshex 009A\string=\glshex 009B\string=\glshex 009C
15626 \string=\glshex 009D\string=\glshex 009E\string=\glshex 009F
15627 }

```

lsxtrspacerules These are space characters.

```

15628 \newcommand*{\glsxtrspacerules}{%
15629 \string' \string'\string;
15630 \string'\glshex 00A0\string'\string;
15631 \string'\glshex 2000\string'\string;
15632 \string'\glshex 2001\string'\string;
15633 \string'\glshex 2002\string'\string;
15634 \string'\glshex 2003\string'\string;
15635 \string'\glshex 2004\string'\string;
15636 \string'\glshex 2005\string'\string;
15637 \string'\glshex 2006\string'\string;
15638 \string'\glshex 2007\string'\string;
15639 \string'\glshex 2008\string'\string;
15640 \string'\glshex 2009\string'\string;
15641 \string'\glshex 200A\string'\string;
15642 \string'\glshex 3000\string'
15643 }

```

nonprintablerules These are non-printable characters (BOM, tabs, line feed and carriage return).

```

15644 \newcommand*{\glsxtrnonprintablerules}{%
15645 \string'\glshex FEFF\string'\string;
15646 \string'\glshex 000A\string'\string;
15647 \string'\glshex 0009\string'\string;
15648 \string'\glshex 000C\string'\string;
15649 \string'\glshex 000B\string'
15650 }

```

gdiacriticrules Combining diacritic marks. This is split into multiple macros.

```

15651 \newcommand*{\glsxtrcombiningdiacriticrules}{%
15652 \glsxtrcombiningdiacriticIrules\string;
15653 \glsxtrcombiningdiacriticIIrules\string;
15654 \glsxtrcombiningdiacriticIIIrules\string;
15655 \glsxtrcombiningdiacriticIVrules
15656 }

```

diacriticIrules First set of combining diacritic marks.

```

15657 \newcommand*{\glsxtrcombiningdiacriticIrules}{%

```

```

15658 \glshex 0301\string;% combining acute
15659 \glshex 0300\string;% combining grave
15660 \glshex 0306\string;% combining breve
15661 \glshex 0302\string;% combining circumflex
15662 \glshex 030C\string;% combining caron
15663 \glshex 030A\string;% combining ring
15664 \glshex 030D\string;% combining vertical line above
15665 \glshex 0308\string;% combining diaeresis
15666 \glshex 030B\string;% combining double acute
15667 \glshex 0303\string;% combining tilde
15668 \glshex 0307\string;% combining dot above
15669 \glshex 0304% combining macron
15670 }

```

iacriticIIrules Second set of combining diacritic marks.

```

15671 \newcommand*{\glxtrcombiningdiacriticIIrules}{%
15672 \glshex 0337\string;% combining short solidus overlay
15673 \glshex 0327\string;% combining cedilla
15674 \glshex 0328\string;% combining ogonek
15675 \glshex 0323\string;% combining dot below
15676 \glshex 0332\string;% combining low line
15677 \glshex 0305\string;% combining overline
15678 \glshex 0309\string;% combining hook above
15679 \glshex 030E\string;% combining double vertical line above
15680 \glshex 030F\string;% combining double grave accent
15681 \glshex 0310\string;% combining candrabindu
15682 \glshex 0311\string;% combining inverted breve
15683 \glshex 0312\string;% combining turned comma above
15684 \glshex 0313\string;% combining comma above
15685 \glshex 0314\string;% combining reversed comma above
15686 \glshex 0315\string;% combining comma above right
15687 \glshex 0316\string;% combining grave accent below
15688 \glshex 0317% combining acute accent below
15689 }

```

acriticIIIrules Third set of combining diacritic marks.

```

15690 \newcommand*{\glxtrcombiningdiacriticIIIrules}{%
15691 \glshex 0318\string;% combining left tack below
15692 \glshex 0319\string;% combining right tack below
15693 \glshex 031A\string;% combining left angle above
15694 \glshex 031B\string;% combining horn
15695 \glshex 031C\string;% combining left half ring below
15696 \glshex 031D\string;% combining up tack below
15697 \glshex 031E\string;% combining down tack below
15698 \glshex 031F\string;% combining plus sign below
15699 \glshex 0320\string;% combining minus sign below
15700 \glshex 0321\string;% combining palatalized hook below
15701 \glshex 0322\string;% combining retroflex hook below
15702 \glshex 0324\string;% combining diaeresis below

```

```

15703 \glshex 0325\string;% combining ring below
15704 \glshex 0326\string;% combining comma below
15705 \glshex 0329\string;% combining vertical line below
15706 \glshex 032A\string;% combining bridge below
15707 \glshex 032B\string;% combining inverted double arch below
15708 \glshex 032C\string;% combining caron below
15709 \glshex 032D\string;% combining circumflex accent below
15710 \glshex 032E\string;% combining breve below
15711 \glshex 032F\string;% combining inverted breve below
15712 \glshex 0330\string;% combining tilde below
15713 \glshex 0331\string;% combining macron below
15714 \glshex 0333\string;% combining double low line
15715 \glshex 0334\string;% combining tilde overlay
15716 \glshex 0335\string;% combining short stroke overlay
15717 \glshex 0336\string;% combining long stroke overlay
15718 \glshex 0338\string;% combining long solidus overlay
15719 \glshex 0339\string;% combining combining right half ring below
15720 \glshex 033A\string;% combining inverted bridge below
15721 \glshex 033B\string;% combining square below
15722 \glshex 033C\string;% combining seagull below
15723 \glshex 033D\string;% combining x above
15724 \glshex 033E\string;% combining vertical tilde
15725 \glshex 033F\string;% combining double overline
15726 \glshex 0342\string;% combining Greek perispomeni
15727 \glshex 0344\string;% combining Greek dialytika tonos
15728 \glshex 0345\string;% combining Greek ypogegrammeni
15729 \glshex 0360\string;% combining double tilde
15730 \glshex 0361\string;% combining double inverted breve
15731 \glshex 0483\string;% combining Cyrillic titlo
15732 \glshex 0484\string;% combining Cyrillic palatalization
15733 \glshex 0485\string;% combining Cyrillic dasia pneumata
15734 \glshex 0486% combining Cyrillic psili pneumata
15735 }

```

iacriticIVrules Fourth set of combining diacritic marks.

```

15736 \newcommand*{\glxtrcombiningdiacriticIVrules}{%
15737 \glshex 20D0\string;% combining left harpoon above
15738 \glshex 20D1\string;% combining right harpoon above
15739 \glshex 20D2\string;% combining long vertical line overlay
15740 \glshex 20D3\string;% combining short vertical line overlay
15741 \glshex 20D4\string;% combining anticlockwise arrow above
15742 \glshex 20D5\string;% combining clockwise arrow above
15743 \glshex 20D6\string;% combining left arrow above
15744 \glshex 20D7\string;% combining right arrow above
15745 \glshex 20D8\string;% combining ring overlay
15746 \glshex 20D9\string;% combining clockwise ring overlay
15747 \glshex 20DA\string;% combining anticlockwise ring overlay
15748 \glshex 20DB\string;% combining three dots above
15749 \glshex 20DC\string;% combining four dots above

```

```

15750 \glshex 20DD\string;% combining enclosing circle
15751 \glshex 20DE\string;% combining enclosing square
15752 \glshex 20DF\string;% combining enclosing diamond
15753 \glshex 20E0\string;% combining enclosing circle backslash
15754 \glshex 20E1% combining left right arrow above
15755 }

```

sxtrhyphenrules Hyphens.

```

15756 \newcommand*{\glsxtrhyphenrules}{%
15757 \string'\string-\string'\string;% ASCII hyphen
15758 \glshex 00AD\string;% soft hyphen
15759 \glshex 2010\string;% hyphen
15760 \glshex 2011\string;% non-breaking hyphen
15761 \glshex 2012\string;% figure dash
15762 \glshex 2013\string;% en dash
15763 \glshex 2014\string;% em dash
15764 \glshex 2015\string;% horizontal bar
15765 \glshex 2212\string=\glshex 207B\string=\glshex 208B% minus sign
15766 }

```

eneralpuncrules General punctuation.

```

15767 \newcommand*{\glsxtrgeneralpuncrules}{%
15768 \glsxtrgeneralpuncIrules
15769 \string<\glsxtrcurrencyrules
15770 \string<\glsxtrgeneralpuncIIrules
15771 }

```

eneralpuncIrules First set of general punctuation.

```

15772 \newcommand*{\glsxtrgeneralpuncIrules}{%
15773 \string'\glshex 005F\string'% underscore
15774 \string<\glshex 00AF% macron
15775 \string<\string'\glshex 002C\string'% comma
15776 \string<\string'\glshex 003B\string'% semi-colon
15777 \string<\string'\glshex 003A\string'% colon
15778 \string<\string'\glshex 0021\string'% exclamation mark
15779 \string<\glshex 00A1% inverted exclamation mark
15780 \string<\string'\glshex 003F\string'% question mark
15781 \string<\glshex 00BF% inverted question mark
15782 \string<\string'\glshex 002F\string'% solidus
15783 \string<\string'\glshex 002E\string'% full stop
15784 \string<\glshex 00B4% acute accent
15785 \string<\string'\glshex 0060\string'% grave accent
15786 \string<\string'\glshex 005E\string'% circumflex accent
15787 \string<\glshex 00A8% diaeresis
15788 \string<\string'\glshex 007E\string'% tilde
15789 \string<\glshex 00B7% middle dot
15790 \string<\glshex 00B8% cedilla
15791 \string<\string'\glshex 0027\string'% straight apostrophe
15792 \string<\string'\glshex 0022\string'% straight double quote

```

```

15793 \string<\glshex 00AB% left guillemet
15794 \string<\glshex 00BB% right guillemet
15795 \string<\string'\glshex 0028\string'% left parenthesis
15796 \string=<\glshex 207D\string=<\glshex 208D% super/subscript left parenthesis
15797 \string<\string'\glshex 0029\string'% right parenthesis
15798 \string=<\glshex 207E\string=<\glshex 208E% super/subscript right parenthesis
15799 \string<\string'\glshex 005B\string'% left square bracket
15800 \string<\string'\glshex 005D\string'% right square bracket
15801 \string<\string'\glshex 007B\string'% left curly bracket
15802 \string<\string'\glshex 007D\string'% right curly bracket
15803 \string<\glshex 00A7% section sign
15804 \string<\glshex 00B6% pilcrow sign
15805 \string<\glshex 00A9% copyright sign
15806 \string<\glshex 00AE% registered sign
15807 \string<\string'\glshex 0040\string'% at sign
15808 }

```

trcurrencyrules General punctuation.

```

15809 \newcommand*{\glxtrcurrencyrules}{%
15810 \glshex 00A4% currency sign
15811 \string<\glshex 0E3F% Thai currency symbol baht
15812 \string<\glshex 00A2% cent sign
15813 \string<\glshex 20A1% colon sign
15814 \string<\glshex 20A2% cruzeiro sign
15815 \string<\string'\glshex 0024\string'% dollar sign
15816 \string<\glshex 20AB% dong sign
15817 \string<\glshex 20AC% euro sign
15818 \string<\glshex 20A3% French franc sign
15819 \string<\glshex 20A4% lira sign
15820 \string<\glshex 20A5% mill sign
15821 \string<\glshex 20A6% naira sign
15822 \string<\glshex 20A7% peseta sign
15823 \string<\glshex 00A3% pound sign
15824 \string<\glshex 20A8% rupee sign
15825 \string<\glshex 20AA% new sheqel sign
15826 \string<\glshex 20A9% won sign
15827 \string<\glshex 00A5% yen sign
15828 }

```

eralpuncIIrules Second set of general punctuation.

```

15829 \newcommand*{\glxtrgeneralpuncIIrules}{%
15830 \string'\glshex 002A\string'% asterisk
15831 \string<\string'\glshex 005C\string'% backslash
15832 \string<\string'\glshex 0026\string'% ampersand
15833 \string<\string'\glshex 0023\string'% hash sign
15834 \string<\string'\glshex 0025\string'% percent sign
15835 \string<\string'\glshex 002B\string'% plus sign
15836 \string=<\glshex 207A\string=<\glshex 208A% super/subscript plus sign
15837 \string<\glshex 00B1% plus-minus sign

```

```

15838 \string<\glshex 00F7% division sign
15839 \string<\glshex 00D7% multiplication sign
15840 \string<\string'\glshex 003C\string'% less-than sign
15841 \string<\string'\glshex 003D\string'% equals sign
15842 \string<\string'\glshex 003E\string'% greater-than sign
15843 \string<\glshex 00AC% not sign
15844 \string<\string'\glshex 007C\string'% vertical bar (pipe)
15845 \string<\glshex 00A6% broken bar
15846 \string<\glshex 00B0% degree sign
15847 \string<\glshex 00B5% micron sign
15848 }

```

eralLatinIrules Basic Latin alphabet.

```

15849 \newcommand*{\glxtrGeneralLatinIrules}{%
15850 \glxtrLatinA
15851 \string<b,B%
15852 \string<c,C%
15853 \string<d,D%
15854 \string<\glxtrLatinE
15855 \string<f,F%
15856 \string<g,G%
15857 \string<\glxtrLatinH
15858 \string<\glxtrLatinI
15859 \string<j,J%
15860 \string<\glxtrLatinK
15861 \string<\glxtrLatinL
15862 \string<\glxtrLatinM
15863 \string<\glxtrLatinN
15864 \string<\glxtrLatinO
15865 \string<\glxtrLatinP
15866 \string<q,Q%
15867 \string<r,R%
15868 \string<\glxtrLatinS
15869 \string<\glxtrLatinT
15870 \string<u,U%
15871 \string<v,V%
15872 \string<w,W%
15873 \string<\glxtrLatinX
15874 \string<y,Y%
15875 \string<z,Z
15876 }

```

ralLatinIIrules General Latin alphabet (eth between D and E, ß treated as SS).

```

15877 \newcommand*{\glxtrGeneralLatinIIrules}{%
15878 \glxtrLatinA
15879 \string<b,B%
15880 \string<c,C%
15881 \string<d,D%
15882 \string<\glxtrLatinEth

```

```

15883 \string<\glxtrLatinE
15884 \string<f,F%
15885 \string<g,G%
15886 \string<\glxtrLatinH
15887 \string<\glxtrLatinI
15888 \string<j,J%
15889 \string<\glxtrLatinK
15890 \string<\glxtrLatinL
15891 \string<\glxtrLatinM
15892 \string<\glxtrLatinN
15893 \string<\glxtrLatinO
15894 \string<\glxtrLatinP
15895 \string<q,Q%
15896 \string<r,R%
15897 \string<\glxtrLatinS
15898 \string& SS \string, \glxtrLatinEszettSs
15899 \string<\glxtrLatinT
15900 \string<u,U%
15901 \string<v,V%
15902 \string<w,W%
15903 \string<\glxtrLatinX
15904 \string<y,Y%
15905 \string<z,Z%
15906 }

```

allLatinIIIrules General Latin alphabet (eth between D and E, ß treated as SZ).

```

15907 \newcommand*{\glxtrGeneralLatinIIIrules}{%
15908 \glxtrLatinA
15909 \string<b,B%
15910 \string<c,C%
15911 \string<d,D%
15912 \string<\glxtrLatinEth
15913 \string<\glxtrLatinE
15914 \string<f,F%
15915 \string<g,G%
15916 \string<\glxtrLatinH
15917 \string<\glxtrLatinI
15918 \string<j,J%
15919 \string<\glxtrLatinK
15920 \string<\glxtrLatinL
15921 \string<\glxtrLatinM
15922 \string<\glxtrLatinN
15923 \string<\glxtrLatinO
15924 \string<\glxtrLatinP
15925 \string<q,Q%
15926 \string<r,R%
15927 \string<\glxtrLatinS
15928 \string& SZ, \glxtrLatinEszettSz
15929 \string<\glxtrLatinT

```



```

15930 \string<u,U%
15931 \string<v,V%
15932 \string<w,W%
15933 \string<\glxtrLatinX
15934 \string<y,Y%
15935 \string<z,Z%
15936 }

```

eralLatinIVrules General Latin alphabet (Æ treated as AE and Ætreated as OE, ptreated as TH, ß treated as SS, eth between D and E).

```

15937 \newcommand*{\glxtrGeneralLatinIVrules}{%
15938 \glxtrLatinA
15939 \string& AE , \glxtrLatinAELigature
15940 \string<b,B%
15941 \string<c,C%
15942 \string<d,D%
15943 \string<\glxtrLatinEth
15944 \string<\glxtrLatinE
15945 \string<f,F%
15946 \string<g,G%
15947 \string<\glxtrLatinH
15948 \string<\glxtrLatinI
15949 \string<j,J%
15950 \string<\glxtrLatinK
15951 \string<\glxtrLatinL
15952 \string<\glxtrLatinM
15953 \string<\glxtrLatinN
15954 \string<\glxtrLatinO
15955 \string& OE , \glxtrLatinOELigature
15956 \string<\glxtrLatinP
15957 \string<q,Q%
15958 \string<r,R%
15959 \string<\glxtrLatinS
15960 \string& SS , \glxtrLatinEszettSs
15961 \string<\glxtrLatinT
15962 \string& th =\glshex 00DE
15963 \string& TH =\glshex 00FE
15964 \string<u,U%
15965 \string<v,V%
15966 \string<w,W%
15967 \string<\glxtrLatinX
15968 \string<y,Y%
15969 \string<z,Z%
15970 }

```

eralLatinVrules General Latin alphabet (eth between D and E, ß treated as SS, Þ treated as TH).

```

15971 \newcommand*{\glxtrGeneralLatinVrules}{%
15972 \glxtrLatinA
15973 \string<b,B%

```

```

15974 \string<c,C%
15975 \string<d,D%
15976 \string<\glxtrLatinEth
15977 \string<\glxtrLatinE
15978 \string<f,F%
15979 \string<g,G%
15980 \string<\glxtrLatinH
15981 \string<\glxtrLatinI
15982 \string<j,J%
15983 \string<\glxtrLatinK
15984 \string<\glxtrLatinL
15985 \string<\glxtrLatinM
15986 \string<\glxtrLatinN
15987 \string<\glxtrLatinO
15988 \string<\glxtrLatinP
15989 \string<q,Q%
15990 \string<r,R%
15991 \string<\glxtrLatinS
15992 \string& SS , \glxtrLatinEszettSs
15993 \string<\glxtrLatinT
15994 \string& th =\glshex 00DE
15995 \string& TH =\glshex 00FE
15996 \string<u,U%
15997 \string<v,V%
15998 \string<w,W%
15999 \string<\glxtrLatinX
16000 \string<y,Y%
16001 \string<z,Z%
16002 }

```

raalLatinVrules General Latin alphabet (eth between D and E, ß treated as SZ, Þ treated as TH).

```

16003 \newcommand*{\glxtrGeneralLatinVrules}{%
16004 \glxtrLatinA
16005 \string<b,B%
16006 \string<c,C%
16007 \string<d,D%
16008 \string<\glxtrLatinEth
16009 \string<\glxtrLatinE
16010 \string<f,F%
16011 \string<g,G%
16012 \string<\glxtrLatinH
16013 \string<\glxtrLatinI
16014 \string<j,J%
16015 \string<\glxtrLatinK
16016 \string<\glxtrLatinL
16017 \string<\glxtrLatinM
16018 \string<\glxtrLatinN
16019 \string<\glxtrLatinO
16020 \string<\glxtrLatinP

```

```

16021 \string<q,Q%
16022 \string<r,R%
16023 \string<\glxtrLatinS
16024 \string& SZ , \glxtrLatinEszettSz
16025 \string<\glxtrLatinT
16026 \string& th =\glshex 00DE
16027 \string& TH =\glshex 00FE
16028 \string<u,U%
16029 \string<v,V%
16030 \string<w,W%
16031 \string<\glxtrLatinX
16032 \string<y,Y%
16033 \string<z,Z%
16034 }

```

alLatinVIIrules General Latin alphabet (Æ between A and B, eth between D and E, insular G as G, CE between O and P, long S equivalent to S, þ between T and U and wynn as W).

```

16035 \newcommand*{\glxtrGeneralLatinVIIrules}{%
16036 \glxtrLatinA
16037 \string<\glxtrLatinAELigature
16038 \string<b,B%
16039 \string<c,C%
16040 \string<d,D%
16041 \string<\glxtrLatinEth
16042 \string<\glxtrLatinE
16043 \string<f,F%
16044 \string<\glxtrLatinInsularG
16045 \string<\glxtrLatinH
16046 \string<\glxtrLatinI
16047 \string<j,J%
16048 \string<\glxtrLatinK
16049 \string<\glxtrLatinL
16050 \string<\glxtrLatinM
16051 \string<\glxtrLatinN
16052 \string<\glxtrLatinO
16053 \string<\glxtrLatinOELigature
16054 \string<\glxtrLatinP
16055 \string<q,Q%
16056 \string<r,R%
16057 \string<\glshex 017F=\glxtrLatinS % s and long s
16058 \string<\glxtrLatinT
16059 \string<\glxtrLatinThorn
16060 \string<u,U%
16061 \string<v,V%
16062 \string< w\string=\glshex 01BF, W\string=\glshex 01F7
16063 \string<\glxtrLatinX
16064 \string<y,Y%
16065 \string<z,Z%
16066 }

```

LatinVIIIrules General Latin alphabet (Æ treated as AE and Ætreated as OE, Þtreated as TH, ß treated as SS, eth treated as D, Ø treated as O, Ł treated as L).

```

16067 \newcommand*{\glxtrGeneralLatinVIIIrules}{%
16068 \glxtrLatinA
16069 \string& AE , \glxtrLatinAELigature
16070 \string<b,B%
16071 \string<c,C%
16072 \string<\glshex 00F0\string;d,\glshex 00D0\string;D% D and eth
16073 \string<\glxtrLatinE
16074 \string<f,F%
16075 \string<g,G%
16076 \string<\glxtrLatinH
16077 \string<\glxtrLatinI
16078 \string<j,J%
16079 \string<\glxtrLatinK
16080 \string<\glshex 0142\string=\glxtrLatinL\string=\glshex 0141% L and \L
16081 \string<\glxtrLatinM
16082 \string<\glxtrLatinN
16083 \string<\glshex 00F8\string=\glxtrLatinO\string=\glshex 00D8% O and \O
16084 \string& OE , \glxtrLatinOELigature
16085 \string<\glxtrLatinP
16086 \string<q,Q%
16087 \string<r,R%
16088 \string<\glxtrLatinS
16089 \string& SS , \glxtrLatinEszettSs
16090 \string<\glxtrLatinT
16091 \string& th =\glshex 00DE
16092 \string& TH =\glshex 00FE
16093 \string<u,U%
16094 \string<v,V%
16095 \string<w,W%
16096 \string<\glxtrLatinX
16097 \string<y,Y%
16098 \string<z,Z%
16099 }

```

\glxtrLatinA

```

16100 \newcommand*{\glxtrLatinA}{%
16101 a\string=\glshex 00AA\string=\glshex 2090,A
16102 }

```

\glxtrLatinE

```

16103 \newcommand*{\glxtrLatinE}{%
16104 e\string=\glshex 2091,E
16105 }

```

\glxtrLatinH

```

16106 \newcommand*{\glxtrLatinH}{%
16107 h\string=\glshex 2095,H

```

16108 }

\glsxtrLatinI

16109 \newcommand*{\glsxtrLatinI}{%

16110 i\string=\glshex 2071,I

16111 }

\glsxtrLatinK

16112 \newcommand*{\glsxtrLatinK}{%

16113 k\string=\glshex 2096,K

16114 }

\glsxtrLatinL

16115 \newcommand*{\glsxtrLatinL}{%

16116 l\string=\glshex 2097,L

16117 }

\glsxtrLatinM

16118 \newcommand*{\glsxtrLatinM}{%

16119 m\string=\glshex 2098,M

16120 }

\glsxtrLatinN

16121 \newcommand*{\glsxtrLatinN}{%

16122 n\string=\glshex 207F\string=\glshex 2099,N

16123 }

\glsxtrLatinO

16124 \newcommand*{\glsxtrLatinO}{%

16125 o\string=\glshex 00BA\string=\glshex 2092,O

16126 }

\glsxtrLatinP

16127 \newcommand*{\glsxtrLatinP}{%

16128 p\string=\glshex 209A,P

16129 }

\glsxtrLatinS

16130 \newcommand*{\glsxtrLatinS}{%

16131 s\string=\glshex 209B,S

16132 }

\glsxtrLatinT

16133 \newcommand*{\glsxtrLatinT}{%

16134 t\string=\glshex 209C,T

16135 }

```

\glxtrLatinX
16136 \newcommand*{\glxtrLatinX}{%
16137   x\string=\glshex 2093,X
16138 }

lsxtrLatinSchwa   Latin schwa (lower case, subscript and upper case).
16139 \newcommand*{\glxtrLatinSchwa}{%
16140   \glshex 0259\string=\glshex 2094,\glshex 018F
16141 }

trLatinEszettSs
16142 \newcommand*{\glxtrLatinEszettSs}{%
16143   \glshex 00DF% eszett
16144   \string=\glshex 017Fs % long S s
16145 }

trLatinEszettSz
16146 \newcommand*{\glxtrLatinEszettSz}{%
16147   \glshex 00DF% eszett
16148   \string= \glshex 017Fz % long S z
16149 }

\glxtrLatinEth
16150 \newcommand*{\glxtrLatinEth}{%
16151   \glshex 00F0,\glshex 00D0% eth
16152 }

lsxtrLatinThorn
16153 \newcommand*{\glxtrLatinThorn}{%
16154   \glshex 00FE,\glshex 00DE% thorn
16155 }

LatinAELigature
16156 \newcommand*{\glxtrLatinAELigature}{%
16157   \glshex 00E6,\glshex 00C6% AE-ligature
16158 }

LatinOELigature
16159 \newcommand*{\glxtrLatinOELigature}{%
16160   \glshex 0153,\glshex 0152% OE-ligature
16161 }

\glxtrLatinAA
16162 \newcommand*{\glxtrLatinAA}{%
16163   \glshex 00E5=a\glshex 030A,% \aa
16164   \glshex 00C5=A\glshex 030A% \AA
16165 }

```

glxtrLatinWynn

```
16166 \newcommand*{\glxtrLatinWynn}{%
16167   \glshex 01BF,\glshex 01F7% wynn
16168 }
```

trLatinInsularG

```
16169 \newcommand*{\glxtrLatinInsularG}{%
16170   \glshex 1D79,\glshex A77D% insular G
16171   \string; g, G
16172 }
```

sxtrLatinOslash

```
16173 \newcommand*{\glxtrLatinOslash}{%
16174   \glshex 00F8,\glshex 00D8% \o, \O
16175 }
```

sxtrLatinLslash

```
16176 \newcommand*{\glxtrLatinLslash}{%
16177   \glshex 0142,\glshex 0141% \l, \L
16178 }
```

thUpGreekIrules Includes digamma between epsilon and zeta.

```
16179 \newcommand*{\glxtrMathUpGreekIrules}{%
16180   \glxtrUpAlpha
16181   \string<\glxtrUpBeta
16182   \string<\glxtrUpGamma
16183   \string<\glxtrUpDelta
16184   \string<\glxtrUpEpsilon
16185   \string<\glxtrUpDigamma
16186   \string<\glxtrUpZeta
16187   \string<\glxtrUpEta
16188   \string<\glxtrUpTheta
16189   \string<\glxtrUpIota
16190   \string<\glxtrUpKappa
16191   \string<\glxtrUpLambda
16192   \string<\glxtrUpMu
16193   \string<\glxtrUpNu
16194   \string<\glxtrUpXi
16195   \string<\glxtrUpOmicron
16196   \string<\glxtrUpPi
16197   \string<\glxtrUpRho
16198   \string<\glxtrUpSigma
16199   \string<\glxtrUpTau
16200   \string<\glxtrUpUpsilon
16201   \string<\glxtrUpPhi
16202   \string<\glxtrUpChi
16203   \string<\glxtrUpPsi
16204   \string<\glxtrUpOmega
16205 }
```

hUpGreekIIrules Doesn't include digamma.

```
16206 \newcommand*{\glxtrMathUpGreekIIrules}{%
16207   \glxtrUpAlpha
16208   \string<\glxtrUpBeta
16209   \string<\glxtrUpGamma
16210   \string<\glxtrUpDelta
16211   \string<\glxtrUpEpsilon
16212   \string<\glxtrUpZeta
16213   \string<\glxtrUpEta
16214   \string<\glxtrUpTheta
16215   \string<\glxtrUpIota
16216   \string<\glxtrUpKappa
16217   \string<\glxtrUpLambda
16218   \string<\glxtrUpMu
16219   \string<\glxtrUpNu
16220   \string<\glxtrUpXi
16221   \string<\glxtrUpOmicron
16222   \string<\glxtrUpPi
16223   \string<\glxtrUpRho
16224   \string<\glxtrUpSigma
16225   \string<\glxtrUpTau
16226   \string<\glxtrUpUpsilon
16227   \string<\glxtrUpPhi
16228   \string<\glxtrUpChi
16229   \string<\glxtrUpPsi
16230   \string<\glxtrUpOmega
16231 }
```

alicGreekIrules Includes (upright) digamma between epsilon and zeta (there isn't an italic digamma), so don't mix with \glxtrMathUpGreekIrules or there may be unexpected results.

```
16232 \newcommand*{\glxtrMathItalicGreekIrules}{%
16233   \glxtrMathItalicAlpha
16234   \string<\glxtrMathItalicBeta
16235   \string<\glxtrMathItalicGamma
16236   \string<\glxtrMathItalicDelta
16237   \string<\glxtrMathItalicEpsilon
16238   \string<\glxtrUpDigamma
16239   \string<\glxtrMathItalicZeta
16240   \string<\glxtrMathItalicEta
16241   \string<\glxtrMathItalicTheta
16242   \string<\glxtrMathItalicIota
16243   \string<\glxtrMathItalicKappa
16244   \string<\glxtrMathItalicLambda
16245   \string<\glxtrMathItalicMu
16246   \string<\glxtrMathItalicNu
16247   \string<\glxtrMathItalicXi
16248   \string<\glxtrMathItalicOmicron
16249   \string<\glxtrMathItalicPi
16250   \string<\glxtrMathItalicRho
```



```

16251 \string<\glxtrMathItalicSigma
16252 \string<\glxtrMathItalicTau
16253 \string<\glxtrMathItalicUpsilon
16254 \string<\glxtrMathItalicPhi
16255 \string<\glxtrMathItalicChi
16256 \string<\glxtrMathItalicPsi
16257 \string<\glxtrMathItalicOmega
16258 }

```

licGreekIIrules Doesn't include digamma.

```

16259 \newcommand*{\glxtrMathItalicGreekIIrules}{%
16260 \glxtrMathItalicAlpha
16261 \string<\glxtrMathItalicBeta
16262 \string<\glxtrMathItalicGamma
16263 \string<\glxtrMathItalicDelta
16264 \string<\glxtrMathItalicEpsilon
16265 \string<\glxtrMathItalicZeta
16266 \string<\glxtrMathItalicEta
16267 \string<\glxtrMathItalicTheta
16268 \string<\glxtrMathItalicIota
16269 \string<\glxtrMathItalicKappa
16270 \string<\glxtrMathItalicLambda
16271 \string<\glxtrMathItalicMu
16272 \string<\glxtrMathItalicNu
16273 \string<\glxtrMathItalicXi
16274 \string<\glxtrMathItalicOmicron
16275 \string<\glxtrMathItalicPi
16276 \string<\glxtrMathItalicRho
16277 \string<\glxtrMathItalicSigma
16278 \string<\glxtrMathItalicTau
16279 \string<\glxtrMathItalicUpsilon
16280 \string<\glxtrMathItalicPhi
16281 \string<\glxtrMathItalicChi
16282 \string<\glxtrMathItalicPsi
16283 \string<\glxtrMathItalicOmega
16284 }

```

pperGreekIrules Upper case only (includes upright digamma).

```

16285 \newcommand*{\glxtrMathItalicUpperGreekIrules}{%
16286 \glshex 1D6E2% upper case alpha (maths italic)
16287 \string<\glshex 1D6E3% upper case beta (maths italic)
16288 \string<\glshex 1D6E4% upper case gamma (maths italic)
16289 \string<\glshex 1D6E5% upper case delta (maths italic)
16290 \string<\glshex 1D6E6% upper case epsilon (maths italic)
16291 \string<\glshex 03DC% upper case digamma
16292 \string<\glshex 1D6E7% upper case zeta (maths italic)
16293 \string<\glshex 1D6E8% upper case eta (maths italic)
16294 \string<\glshex 1D6E9% upper case theta (maths italic)
16295 \string=<\glshex 1D6F3% upper case theta variant (maths italic)

```

```

16296 \string<\glshex 1D6EA% upper case iota (maths italic)
16297 \string<\glshex 1D6EB% upper case kappa (maths italic)
16298 \string<\glshex 1D6EC% upper case lambda (maths italic)
16299 \string<\glshex 1D6ED% upper case mu (maths italic)
16300 \string<\glshex 1D6EE% upper case nu (maths italic)
16301 \string<\glshex 1D6EF% upper case xi (maths italic)
16302 \string<\glshex 1D6F0% upper case omicron (maths italic)
16303 \string<\glshex 1D6F1% upper case pi (maths italic)
16304 \string<\glshex 1D6F2% upper case rho (maths italic)
16305 \string<\glshex 1D6F4% upper case sigma (maths italic)
16306 \string<\glshex 1D6F5% upper case tau (maths italic)
16307 \string<\glshex 1D6F6% upper case upsilon (maths italic)
16308 \string<\glshex 1D6F7% upper case phi (maths italic)
16309 \string<\glshex 1D6F8% upper case chi (maths italic)
16310 \string<\glshex 1D6F9% upper case psi (maths italic)
16311 \string<\glshex 1D6FA% upper case omega (maths italic)
16312 }

```

perGreekIIrules Upper case only (doesn't include upright digamma).

```

16313 \newcommand*{\glxtrMathItalicUpperGreekIIrules}{%
16314 \glshex 1D6E2% upper case alpha (maths italic)
16315 \string<\glshex 1D6E3% upper case beta (maths italic)
16316 \string<\glshex 1D6E4% upper case gamma (maths italic)
16317 \string<\glshex 1D6E5% upper case delta (maths italic)
16318 \string<\glshex 1D6E6% upper case epsilon (maths italic)
16319 \string<\glshex 1D6E7% upper case zeta (maths italic)
16320 \string<\glshex 1D6E8% upper case eta (maths italic)
16321 \string<\glshex 1D6E9% upper case theta (maths italic)
16322 \string<\glshex 1D6F3% upper case theta variant (maths italic)
16323 \string<\glshex 1D6EA% upper case iota (maths italic)
16324 \string<\glshex 1D6EB% upper case kappa (maths italic)
16325 \string<\glshex 1D6EC% upper case lambda (maths italic)
16326 \string<\glshex 1D6ED% upper case mu (maths italic)
16327 \string<\glshex 1D6EE% upper case nu (maths italic)
16328 \string<\glshex 1D6EF% upper case xi (maths italic)
16329 \string<\glshex 1D6F0% upper case omicron (maths italic)
16330 \string<\glshex 1D6F1% upper case pi (maths italic)
16331 \string<\glshex 1D6F2% upper case rho (maths italic)
16332 \string<\glshex 1D6F4% upper case sigma (maths italic)
16333 \string<\glshex 1D6F5% upper case tau (maths italic)
16334 \string<\glshex 1D6F6% upper case upsilon (maths italic)
16335 \string<\glshex 1D6F7% upper case phi (maths italic)
16336 \string<\glshex 1D6F8% upper case chi (maths italic)
16337 \string<\glshex 1D6F9% upper case psi (maths italic)
16338 \string<\glshex 1D6FA% upper case omega (maths italic)
16339 }

```

lowerGreekIrules Lower case only (includes upright digamma).

```

16340 \newcommand*{\glxtrMathItalicLowerGreekIrules}{%

```

```

16341 \glshex 1D6FC% lower case alpha (maths italic)
16342 \string<\glshex 1D6FD% lower case beta (maths italic)
16343 \string<\glshex 1D6FE% lower case gamma (maths italic)
16344 \string<\glshex 1D6FF% lower case delta (maths italic)
16345 \string<\glshex 1D700% lower case epsilon (maths italic)
16346 \string=\glshex 1D716% lower case epsilon variant (maths italic)
16347 \string<\glshex 03DD% lower case digamma
16348 \string<\glshex 1D701% lower case zeta (maths italic)
16349 \string<\glshex 1D702% lower case eta (maths italic)
16350 \string<\glshex 1D703% lower case theta (maths italic)
16351 \string=\glshex 1D717% lower case theta variant (maths italic)
16352 \string<\glshex 1D704% lower case iota (maths italic)
16353 \string<\glshex 1D705% lower case kappa (maths italic)
16354 \string=\glshex 1D718% lower case kappa variant (maths italic)
16355 \string<\glshex 1D706% lower case lambda (maths italic)
16356 \string<\glshex 1D707% lower case mu (maths italic)
16357 \string<\glshex 1D708% lower case nu (maths italic)
16358 \string<\glshex 1D709% lower case xi (maths italic)
16359 \string<\glshex 1D70A% lower case omicron (maths italic)
16360 \string<\glshex 1D70B% lower case pi (maths italic)
16361 \string=\glshex 1D71B% lower case pi variant (maths italic)
16362 \string<\glshex 1D70C% lower case rho (maths italic)
16363 \string=\glshex 1D71A% lower case rho variant (maths italic)
16364 \string<\glshex 1D70D% lower case final sigma (maths italic)
16365 \string=\glshex 1D70E% lower case sigma (maths italic)
16366 \string<\glshex 1D70F% lower case tau (maths italic)
16367 \string<\glshex 1D710% lower case upsilon (maths italic)
16368 \string<\glshex 1D711% lower case phi (maths italic)
16369 \string=\glshex 1D719% lower case phi variant (maths italic)
16370 \string<\glshex 1D712% lower case chi (maths italic)
16371 \string<\glshex 1D713% lower case psi (maths italic)
16372 \string<\glshex 1D714% lower case omega (maths italic)
16373 }

```

LowerGreeklIrules Lower case only (doesn't includes upright digamma).

```

16374 \newcommand*{\glxtrMathItalicLowerGreeklIrules}{%
16375 \glshex 1D6FC% lower case alpha (maths italic)
16376 \string<\glshex 1D6FD% lower case beta (maths italic)
16377 \string<\glshex 1D6FE% lower case gamma (maths italic)
16378 \string<\glshex 1D6FF% lower case delta (maths italic)
16379 \string<\glshex 1D700% lower case epsilon (maths italic)
16380 \string=\glshex 1D716% lower case epsilon variant (maths italic)
16381 \string<\glshex 1D701% lower case zeta (maths italic)
16382 \string<\glshex 1D702% lower case eta (maths italic)
16383 \string<\glshex 1D703% lower case theta (maths italic)
16384 \string=\glshex 1D717% lower case theta variant (maths italic)
16385 \string<\glshex 1D704% lower case iota (maths italic)
16386 \string<\glshex 1D705% lower case kappa (maths italic)
16387 \string=\glshex 1D718% lower case kappa variant (maths italic)

```

```

16388 \string<\glshex 1D706% lower case lambda (maths italic)
16389 \string<\glshex 1D707% lower case mu (maths italic)
16390 \string<\glshex 1D708% lower case nu (maths italic)
16391 \string<\glshex 1D709% lower case xi (maths italic)
16392 \string<\glshex 1D70A% lower case omicron (maths italic)
16393 \string<\glshex 1D70B% lower case pi (maths italic)
16394 \string=<\glshex 1D71B% lower case pi variant (maths italic)
16395 \string<\glshex 1D70C% lower case rho (maths italic)
16396 \string=<\glshex 1D71A% lower case rho variant (maths italic)
16397 \string<\glshex 1D70D% lower case final sigma (maths italic)
16398 \string=<\glshex 1D70E% lower case sigma (maths italic)
16399 \string<\glshex 1D70F% lower case tau (maths italic)
16400 \string<\glshex 1D710% lower case upsilon (maths italic)
16401 \string<\glshex 1D711% lower case phi (maths italic)
16402 \string=<\glshex 1D719% lower case phi variant (maths italic)
16403 \string<\glshex 1D712% lower case chi (maths italic)
16404 \string<\glshex 1D713% lower case psi (maths italic)
16405 \string<\glshex 1D714% lower case omega (maths italic)
16406 }

```

MathGreekIrules Includes both upright and italic with digamma between epsilon and zeta.

```

16407 \newcommand*{\glxtrMathGreekIrules}{%
16408 \glxtrMathItalicAlpha
16409 \string;\glxtrUpAlpha
16410 \string<\glxtrMathItalicBeta
16411 \string;\glxtrUpBeta
16412 \string<\glxtrMathItalicGamma
16413 \string;\glxtrUpGamma
16414 \string<\glxtrMathItalicDelta
16415 \string;\glxtrUpDelta
16416 \string<\glxtrMathItalicEpsilon
16417 \string;\glxtrUpEpsilon
16418 \string<\glxtrUpDigamma
16419 \string<\glxtrMathItalicZeta
16420 \string;\glxtrUpZeta
16421 \string<\glxtrMathItalicEta
16422 \string;\glxtrUpEta
16423 \string<\glxtrMathItalicTheta
16424 \string;\glxtrUpTheta
16425 \string<\glxtrMathItalicIota
16426 \string;\glxtrUpIota
16427 \string<\glxtrMathItalicKappa
16428 \string;\glxtrUpKappa
16429 \string<\glxtrMathItalicLambda
16430 \string;\glxtrUpLambda
16431 \string<\glxtrMathItalicMu
16432 \string;\glxtrUpMu
16433 \string<\glxtrMathItalicNu
16434 \string;\glxtrUpNu

```

```

16435 \string<\glxtrMathItalicXi
16436 \string;\glxtrUpXi
16437 \string<\glxtrMathItalicOmicron
16438 \string;\glxtrUpOmicron
16439 \string<\glxtrMathItalicPi
16440 \string;\glxtrUpPi
16441 \string<\glxtrMathItalicRho
16442 \string;\glxtrUpRho
16443 \string<\glxtrMathItalicSigma
16444 \string;\glxtrUpSigma
16445 \string<\glxtrMathItalicTau
16446 \string;\glxtrUpTau
16447 \string<\glxtrMathItalicUpsilon
16448 \string;\glxtrUpUpsilon
16449 \string<\glxtrMathItalicPhi
16450 \string;\glxtrUpPhi
16451 \string<\glxtrMathItalicChi
16452 \string;\glxtrUpChi
16453 \string<\glxtrMathItalicPsi
16454 \string;\glxtrUpPsi
16455 \string<\glxtrMathItalicOmega
16456 \string;\glxtrUpOmega
16457 }

```

mathGreekIIrules Includes both upright and italic (digamma not included).

```

16458 \newcommand*{\glxtrMathGreekIIrules}{%
16459 \glxtrMathItalicAlpha
16460 \string;\glxtrUpAlpha
16461 \string<\glxtrMathItalicBeta
16462 \string;\glxtrUpBeta
16463 \string<\glxtrMathItalicGamma
16464 \string;\glxtrUpGamma
16465 \string<\glxtrMathItalicDelta
16466 \string;\glxtrUpDelta
16467 \string<\glxtrMathItalicEpsilon
16468 \string;\glxtrUpEpsilon
16469 \string<\glxtrMathItalicZeta
16470 \string;\glxtrUpZeta
16471 \string<\glxtrMathItalicEta
16472 \string;\glxtrUpEta
16473 \string<\glxtrMathItalicTheta
16474 \string;\glxtrUpTheta
16475 \string<\glxtrMathItalicIota
16476 \string;\glxtrUpIota
16477 \string<\glxtrMathItalicKappa
16478 \string;\glxtrUpKappa
16479 \string<\glxtrMathItalicLambda
16480 \string;\glxtrUpLambda
16481 \string<\glxtrMathItalicMu

```

```

16482 \string;\glxtrUpMu
16483 \string<\glxtrMathItalicNu
16484 \string;\glxtrUpNu
16485 \string<\glxtrMathItalicXi
16486 \string;\glxtrUpXi
16487 \string<\glxtrMathItalicOmicron
16488 \string;\glxtrUpOmicron
16489 \string<\glxtrMathItalicPi
16490 \string;\glxtrUpPi
16491 \string<\glxtrMathItalicRho
16492 \string;\glxtrUpRho
16493 \string<\glxtrMathItalicSigma
16494 \string;\glxtrUpSigma
16495 \string<\glxtrMathItalicTau
16496 \string;\glxtrUpTau
16497 \string<\glxtrMathItalicUpsilon
16498 \string;\glxtrUpUpsilon
16499 \string<\glxtrMathItalicPhi
16500 \string;\glxtrUpPhi
16501 \string<\glxtrMathItalicChi
16502 \string;\glxtrUpChi
16503 \string<\glxtrMathItalicPsi
16504 \string;\glxtrUpPsi
16505 \string<\glxtrMathItalicOmega
16506 \string;\glxtrUpOmega
16507 }

```

\glxtrUpAlpha

```

16508 \newcommand*{\glxtrUpAlpha}{%
16509 \glshex 03B1,% lower case alpha
16510 \glshex 0391% upper case alpha
16511 }

```

\glxtrUpBeta

```

16512 \newcommand*{\glxtrUpBeta}{%
16513 \glshex 03B2,% lower case beta
16514 \glshex 0392% upper case beta
16515 }

```

\glxtrUpGamma

```

16516 \newcommand*{\glxtrUpGamma}{%
16517 \glshex 03B3,% lower case gamma
16518 \glshex 0393% upper case gamma
16519 }

```

\glxtrUpDelta

```

16520 \newcommand*{\glxtrUpDelta}{%
16521 \glshex 03B4,% lower case delta
16522 \glshex 0394% upper case delta

```

16523 }

glsxtrUpEpsilon

```
16524 \newcommand*{\glsxtrUpEpsilon}{%
16525   \glsheX 03B5% lower case epsilon
16526   \string=\glsheX 03F5,% lower case epsilon variant
16527   \glsheX 0395% upper case epsilon
16528 }
```

glsxtrUpDigamma

```
16529 \newcommand*{\glsxtrUpDigamma}{%
16530   \glsheX 03DD,% lower case digamma
16531   \glsheX 03DC% upper case digamma
16532 }
```

\glsxtrUpZeta

```
16533 \newcommand*{\glsxtrUpZeta}{%
16534   \glsheX 03B6,% lower case zeta
16535   \glsheX 0396% upper case zeta
16536 }
```

\glsxtrUpEta

```
16537 \newcommand*{\glsxtrUpEta}{%
16538   \glsheX 03B7,% lower case eta
16539   \glsheX 0397% upper case eta
16540 }
```

\glsxtrUpTheta

```
16541 \newcommand*{\glsxtrUpTheta}{%
16542   \glsheX 03B8% lower case theta
16543   \string=\glsheX 03D1,% lower case theta variant
16544   \glsheX 0398% upper case theta
16545 }
```

\glsxtrUpIota

```
16546 \newcommand*{\glsxtrUpIota}{%
16547   \glsheX 03B9,% lower case iota
16548   \glsheX 0399% upper case iota
16549 }
```

\glsxtrUpKappa

```
16550 \newcommand*{\glsxtrUpKappa}{%
16551   \glsheX 03BA% lower case kappa
16552   \string=\glsheX 03F0,% lower case kappa variant
16553   \glsheX 039A% upper case kappa
16554 }
```

\glxtrUpLambda

```
16555 \newcommand*{\glxtrUpLambda}{%  
16556 \glshex 03BB,% lower lambda  
16557 \glshex 039B% upper case lambda  
16558 }
```

\glxtrUpMu

```
16559 \newcommand*{\glxtrUpMu}{%  
16560 \glshex 03BC,% lower case mu  
16561 \glshex 039C% upper case mu  
16562 }
```

\glxtrUpNu

```
16563 \newcommand*{\glxtrUpNu}{%  
16564 \glshex 03BD,% lower case nu  
16565 \glshex 039D% upper case nu  
16566 }
```

\glxtrUpXi

```
16567 \newcommand*{\glxtrUpXi}{%  
16568 \glshex 03BE,% lower case xi  
16569 \glshex 039E% upper case xi  
16570 }
```

\glxtrUpOmicron

```
16571 \newcommand*{\glxtrUpOmicron}{%  
16572 \glshex 03BF,% lower case omicron  
16573 \glshex 039F% upper case omicron  
16574 }
```

\glxtrUpPi

```
16575 \newcommand*{\glxtrUpPi}{%  
16576 \glshex 03C0% lower case pi  
16577 \string=\glshex 03D6,% lower case pi variant  
16578 \glshex 03A0% upper case pi  
16579 }
```

\glxtrUpRho

```
16580 \newcommand*{\glxtrUpRho}{%  
16581 \glshex 03C1% lower case rho  
16582 \string=\glshex 03F1,% lower case rho variant  
16583 \glshex 03A1% upper case rho  
16584 }
```

\glxtrUpSigma

```
16585 \newcommand*{\glxtrUpSigma}{%  
16586 \glshex 03C2% lower case sigma  
16587 \string=\glshex 03C3,% lower case sigma
```



```
16588 \glshex 03A3% upper case sigma
16589 }
```

\glxtrUpTau

```
16590 \newcommand*{\glxtrUpTau}{%
16591 \glshex 03C4,% lower case tau
16592 \glshex 03A4% upper case tau
16593 }
```

glxtrUpUpsilon

```
16594 \newcommand*{\glxtrUpUpsilon}{%
16595 \glshex 03C5,% lower case upsilon
16596 \glshex 03A5% upper case upsilon
16597 }
```

\glxtrUpPhi

```
16598 \newcommand*{\glxtrUpPhi}{%
16599 \glshex 03C6% lower case phi
16600 \string=\glshex 03D5,% lower case phi variant
16601 \glshex 03A6% upper case phi
16602 }
```

\glxtrUpChi

```
16603 \newcommand*{\glxtrUpChi}{%
16604 \glshex 03C7,% lower case chi
16605 \glshex 03A7% upper case chi
16606 }
```

\glxtrUpPsi

```
16607 \newcommand*{\glxtrUpPsi}{%
16608 \glshex 03C8,% lower case psi
16609 \glshex 03A8% upper case psi
16610 }
```

\glxtrUpOmega

```
16611 \newcommand*{\glxtrUpOmega}{%
16612 \glshex 03C9,% lower case omega
16613 \glshex 03A9% upper case omega
16614 }
```

MathItalicAlpha

```
16615 \newcommand*{\glxtrMathItalicAlpha}{%
16616 \glshex 1D6FC,% lower case alpha (maths italic)
16617 \glshex 1D6E2% upper case alpha (maths italic)
16618 }
```

rMathItalicBeta

```
16619 \newcommand*{\glxtrMathItalicBeta}{%
```

```

16620 \glshex 1D6FD,% lower case beta (maths italic)
16621 \glshex 1D6E3% upper case beta (maths italic)
16622 }

```

MathItalicGamma

```

16623 \newcommand*{\glxtrMathItalicGamma}{%
16624 \glshex 1D6FE,% lower case gamma (maths italic)
16625 \glshex 1D6E4% upper case gamma (maths italic)
16626 }

```

MathItalicDelta

```

16627 \newcommand*{\glxtrMathItalicDelta}{%
16628 \glshex 1D6FF,% lower case delta (maths italic)
16629 \glshex 1D6E5% upper case delta (maths italic)
16630 }

```

MathItalicEpsilon

```

16631 \newcommand*{\glxtrMathItalicEpsilon}{%
16632 \glshex 1D700% lower case epsilon (maths italic)
16633 \string=\glshex 1D716,% lower case epsilon variant (maths italic)
16634 \glshex 1D6E6% upper case epsilon (maths italic)
16635 }

```

MathItalicZeta

```

16636 \newcommand*{\glxtrMathItalicZeta}{%
16637 \glshex 1D701,% lower case zeta (maths italic)
16638 \glshex 1D6E7% upper case zeta (maths italic)
16639 }

```

MathItalicEta

```

16640 \newcommand*{\glxtrMathItalicEta}{%
16641 \glshex 1D702,% lower case eta (maths italic)
16642 \glshex 1D6E8% upper case eta (maths italic)
16643 }

```

MathItalicTheta

```

16644 \newcommand*{\glxtrMathItalicTheta}{%
16645 \glshex 1D703% lower case theta (maths italic)
16646 \string=\glshex 1D717,% lower case theta variant (maths italic)
16647 \glshex 1D6E9% upper case theta (maths italic)
16648 \string=\glshex 1D6F3% upper case theta variant (maths italic)
16649 }

```

MathItalicIota

```

16650 \newcommand*{\glxtrMathItalicIota}{%
16651 \glshex 1D704,% lower case iota (maths italic)
16652 \glshex 1D6EA% upper case iota (maths italic)
16653 }

```

MathItalicKappa

```
16654 \newcommand*{\glxtrMathItalicKappa}{%
16655   \glshex 1D705% lower case kappa (maths italic)
16656   \string=\glshex 1D718,% lower case kappa variant (maths italic)
16657   \glshex 1D6EB% upper case kappa (maths italic)
16658 }
```

athItalicLambda

```
16659 \newcommand*{\glxtrMathItalicLambda}{%
16660   \glshex 1D706,% lower case lambda (maths italic)
16661   \glshex 1D6EC% upper case lambda (maths italic)
16662 }
```

xtrMathItalicMu

```
16663 \newcommand*{\glxtrMathItalicMu}{%
16664   \glshex 1D707,% lower case mu (maths italic)
16665   \glshex 1D6ED% upper case mu (maths italic)
16666 }
```

xtrMathItalicNu

```
16667 \newcommand*{\glxtrMathItalicNu}{%
16668   \glshex 1D708,% lower case nu (maths italic)
16669   \glshex 1D6EE% upper case nu (maths italic)
16670 }
```

xtrMathItalicXi

```
16671 \newcommand*{\glxtrMathItalicXi}{%
16672   \glshex 1D709,% lower case xi (maths italic)
16673   \glshex 1D6EF% upper case xi (maths italic)
16674 }
```

thItalicOmicron

```
16675 \newcommand*{\glxtrMathItalicOmicron}{%
16676   \glshex 1D70A,% lower case omicron (maths italic)
16677   \glshex 1D6F0% upper case omicron (maths italic)
16678 }
```

xtrMathItalicPi

```
16679 \newcommand*{\glxtrMathItalicPi}{%
16680   \glshex 1D70B% lower case pi (maths italic)
16681   \string=\glshex 1D71B,% lower case pi variant (maths italic)
16682   \glshex 1D6F1% upper case pi (maths italic)
16683 }
```

trMathItalicRho

```
16684 \newcommand*{\glxtrMathItalicRho}{%
16685   \glshex 1D70C% lower case rho (maths italic)
16686   \string=\glshex 1D71A,% lower case rho variant (maths italic)
```

```

16687 \glshex 1D6F2% upper case rho (maths italic)
16688 }

```

MathItalicSigma

```

16689 \newcommand*{\glsxtrMathItalicSigma}{%
16690 \glshex 1D70D% lower case final sigma (maths italic)
16691 \string=\glshex 1D70E,% lower case sigma (maths italic)
16692 \glshex 1D6F4% upper case sigma (maths italic)
16693 }

```

trMathItalicTau

```

16694 \newcommand*{\glsxtrMathItalicTau}{%
16695 \glshex 1D70F,% lower case tau (maths italic)
16696 \glshex 1D6F5% upper case tau (maths italic)
16697 }

```

thItalicUpsilon

```

16698 \newcommand*{\glsxtrMathItalicUpsilon}{%
16699 \glshex 1D710,% lower case upsilon (maths italic)
16700 \glshex 1D6F6% upper case upsilon (maths italic)
16701 }

```

trMathItalicPhi

```

16702 \newcommand*{\glsxtrMathItalicPhi}{%
16703 \glshex 1D711% lower case phi (maths italic)
16704 \string=\glshex 1D719,% lower case phi variant (maths italic)
16705 \glshex 1D6F7% upper case phi (maths italic)
16706 }

```

trMathItalicChi

```

16707 \newcommand*{\glsxtrMathItalicChi}{%
16708 \glshex 1D712,% lower case chi (maths italic)
16709 \glshex 1D6F8% upper case chi (maths italic)
16710 }

```

trMathItalicPsi

```

16711 \newcommand*{\glsxtrMathItalicPsi}{%
16712 \glshex 1D713,% lower case psi (maths italic)
16713 \glshex 1D6F9% upper case psi (maths italic)
16714 }

```

MathItalicOmega

```

16715 \newcommand*{\glsxtrMathItalicOmega}{%
16716 \glshex 1D714,% lower case omega (maths italic)
16717 \glshex 1D6FA% upper case omega (maths italic)
16718 }

```

thItalicPartial

```
16719 \newcommand*{\glxtrMathItalicPartial}{%
16720 \glshex 1D715% partial differential (maths italic)
16721 }
```

MathItalicNabla

```
16722 \newcommand*{\glxtrMathItalicNabla}{%
16723 \glshex 1D6FB% nabla (maths italic)
16724 }
```

lsxtrdigitrules Digits from the Basic Latin set and subscript and superscript digit rules.

```
16725 \newcommand*{\glxtrdigitrules}{%
16726 0\string=\glshex 2080\string=\glshex 2070
16727 \string<1\string=\glshex 2081\string=\glshex 00B9
16728 \string<2\string=\glshex 2082\string=\glshex 00B2
16729 \string<3\string=\glshex 2083\string=\glshex 00B3
16730 \string<4\string=\glshex 2084\string=\glshex 2074
16731 \string<5\string=\glshex 2085\string=\glshex 2075
16732 \string<6\string=\glshex 2086\string=\glshex 2076
16733 \string<7\string=\glshex 2087\string=\glshex 2077
16734 \string<8\string=\glshex 2088\string=\glshex 2078
16735 \string<9\string=\glshex 2089\string=\glshex 2079
16736 }
```

BasicDigitrules Digits from the Basic Latin set.

```
16737 \newcommand*{\glxtrBasicDigitrules}{%
16738 0\string<1\string<2\string<3\string<4%
16739 \string<5\string<6\string<7\string<8\string<9%
16740 }
```

criptDigitrules Subscript digits.

```
16741 \newcommand*{\glxtrSubScriptDigitrules}{%
16742 \glshex 2080% subscript 0
16743 \string<\glshex 2081% subscript 1
16744 \string<\glshex 2082% subscript 2
16745 \string<\glshex 2083% subscript 3
16746 \string<\glshex 2084% subscript 4
16747 \string<\glshex 2085% subscript 5
16748 \string<\glshex 2086% subscript 6
16749 \string<\glshex 2087% subscript 7
16750 \string<\glshex 2088% subscript 8
16751 \string<\glshex 2089% subscript 9
16752 }
```

criptDigitrules Superscript digits.

```
16753 \newcommand*{\glxtrSuperScriptDigitrules}{%
16754 \glshex 2070% superscript 0
16755 \string<\glshex 00B9% superscript 1
```

```

16756 \string<\glshex 00B2% superscript 2
16757 \string<\glshex 00B3% superscript 3
16758 \string<\glshex 2074% superscript 4
16759 \string<\glshex 2075% superscript 5
16760 \string<\glshex 2076% superscript 6
16761 \string<\glshex 2077% superscript 7
16762 \string<\glshex 2078% superscript 8
16763 \string<\glshex 2079% superscript 9
16764 }

```

trfractionrules Vulgar fractions.

```

16765 \newcommand*{\glxtrfractionrules}{%
16766 \glshex 215F% fraction numerator one (1/)
16767 \string<\glshex 2189% zero thirds (0/3 = 0)
16768 \string<\glshex 2152% one tenth (1/10 = 0.1)
16769 \string<\glshex 2151% one ninth (1/9 ~ 0.111)
16770 \string<\glshex 215B% one eighth (1/8 = 0.125)
16771 \string<\glshex 2150% one seventh (1/7 ~ 0.143)
16772 \string<\glshex 2159% one sixth (1/6 ~ 0.167)
16773 \string<\glshex 2155% one fifth (1/5 = 0.2)
16774 \string<\glshex 00BC% one quarter (1/4 = 0.25)
16775 \string<\glshex 2153% one third (1/3 ~ 0.333)
16776 \string<\glshex 215C% three eighths (3/8 = 0.375)
16777 \string<\glshex 2156% two fifths (2/5 = 0.4)
16778 \string<\glshex 00BD% one half (1/2 = 0.5)
16779 \string<\glshex 2157% three fifths (3/5 = 0.6)
16780 \string<\glshex 215D% five eighths (5/8 = 0.625)
16781 \string<\glshex 2154% two thirds (2/3 ~ 0.667)
16782 \string<\glshex 00BE% three quarters (3/4 = 0.75)
16783 \string<\glshex 2158% four fifths (4/5 = 0.8)
16784 \string<\glshex 215A% five sixths (5/6 ~ 0.833)
16785 \string<\glshex 215E% seven eighths (7/8 = 0.875)
16786 }

```

sxtrdialecthook Check for scripts associated with the document dialects.

```

16787 \renewcommand{\@glxtrdialecthook}{%
16788 \ifundef\CurrentTrackedScript
16789 {%
16790 \TrackLangIfHasDefaultScript{\CurrentTrackedLanguage}%
16791 {%
16792 \edef\CurrentTrackedScript{%
16793 \TrackLangGetDefaultScript\CurrentTrackedLanguage}%
16794 }%
16795 {}%
16796 }%
16797 {}%
16798 \ifdef\CurrentTrackedScript
16799 {%
16800 \let\gls@orgTrackLangRequireDialectPrefix\TrackLangRequireDialectPrefix

```

```

16801 \def\TrackLangRequireDialectPrefix{glossariesxtr-}%
16802 \let\CurrentTrackedTag\CurrentTrackedScript
16803 \IfFileExists{\TrackLangRequireDialectPrefix\CurrentTrackedTag.ldf}
16804 {\RequireGlossariesExtraLang{\CurrentTrackedTag}}%
16805 {}%
16806 \let\TrackLangRequireDialectPrefix\gls@orgTrackLangRequireDialectPrefix
16807 }%
16808 {}%
16809 }

```

If `\glsxtr@loaddialect` has been defined, then `glossaries-extra-bib2gls` has been loaded after `glossaries-extra`. (For example, through `\glossariesextrasetup`.) Not recommended, but if this has been done try to find the associated language resources.

```

16810 \ifdef\glsxtr@loaddialect
16811 {%
16812 \ifpackageloaded{tracklang}
16813 {%
16814 \AnyTrackedLanguages
16815 {%
16816 \ForEachTrackedDialect{\this@dialect}{\glsxtr@loaddialect}%
16817 }%
16818 {}%
16819 }
16820 {}
16821 }
16822 {}

```

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:

```
16823 \NeedsTeXFormat{LaTeX2e}
```

```
16824 \ProvidesPackage{glossaries-extra-stylemods}[2021/11/22 v1.48 (NLCT)]
```

Provide package options to automatically load required predefined styles. The simplest method is to just test for the existence of the file `glossary-<option>.sty`. Packages can't be loaded whilst the options are being processed, so save the list in `\@glxtr@loadstyles`.

```
sxtr@loadstyles
```

```
16825 \newcommand*{\@glxtr@loadstyles}{}
```

all Provide all known styles.

```
16826 \DeclareOption{all}{%
```

```
16827   \appto\@glxtr@loadstyles{%
```

```
16828     \RequirePackage{glossary-inline}%
```

```
16829     \RequirePackage{glossary-list}%
```

```
16830     \RequirePackage{glossary-tree}%
```

```
16831     \RequirePackage{glossary-mcols}%
```

```
16832     \RequirePackage{glossary-long}%
```

```
16833     \RequirePackage{glossary-longragged}%
```

```
16834     \RequirePackage{glossary-longbooktabs}%
```

```
16835     \RequirePackage{glossary-super}%
```

```
16836     \RequirePackage{glossary-superragged}%
```

```
16837     \RequirePackage{glossary-bookindex}%
```

```
16838     \RequirePackage{glossary-longextra}%
```

```
16839     \RequirePackage{glossary-topic}%
```

```
16840   }
```

```
16841 }
```

```
16842 \DeclareOption*{%
```

```
16843   \IfFileExists{glossary-\CurrentOption.sty}
```

```
16844   {\eappto\@glxtr@loadstyles{%
```

```
16845     \noexpand\RequirePackage{glossary-\CurrentOption}}%
```

```
16846   }%
```



```

16847   {%
16848     \PackageError{glossaries-extra-styles}%
16849     {Unknown option ‘\CurrentOption’}{}%
16850   }%
16851 }

```

Process the package options:

```
16852 \ProcessOptions
```

Load the required packages:

```
16853 \@glsxtr@loadstyles
```

Adjust the styles so that they all have the post description hook. Also, instead of having a hard-coded \space before the location, use:

sxtrprelocation This uses \providecommand as the same command is also provided by glossary-bookindex.

```
16854 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

ewglossarystyle

```

16855 \providecommand{\renewglossarystyle}[2]{%
16856   \ifcsundef{@glsstyle@#1}%
16857   {%
16858     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
16859   }%
16860   {%
16861     \csdef{@glsstyle@#1}{#2}%
16862   }%
16863 }

```

2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying to add this.

```

16864 \ifdef{\@glsstyle@listdotted}
16865 {%
16866   \renewglossarystyle{listdotted}{%
16867     \setglossarystyle{list}%
16868     \renewcommand*{\glossentry}[2]{%
16869       \item[]\makebox[\glslistdottedwidth][l]{%
16870         \glsentryitem{##1}%
16871         \glstarget{##1}{\glossentryname{##1}}%
16872         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
16873         \glossentrydesc{##1}\glspostdescription}%
16874     \renewcommand*{\subglossentry}[3]{%
16875       \item[]\makebox[\glslistdottedwidth][l]{%
16876         \glssubentryitem{##2}%
16877         \glstarget{##2}{\glossentryname{##2}}%
16878         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%

```

```

16879     \glossentrydesc{##2}\glspostdescription}%
16880 }
16881 }
16882 {%

```

Assume the style isn't required if it hasn't already been defined.

```

16883 }

```

The sublistdotted style doesn't display the description for top-level entries. Sub-level entries use the listdottedstyle.

The other list styles would be easier to adapt if the space before the number list wasn't hard coded.

```

16884 \ifdef{\@glsstyle@list}
16885 {%

```

listprelocation Space before number list for top-level entries.

```

16886 \newcommand{\glslistprelocation}{\glstrprelocation}

```

childprelocation Space before number list for child entries.

```

16887 \newcommand{\glslistchildprelocation}{\glslistprelocation}

```

childpostlocation Full stop after number list.

```

16888 \newcommand{\glslistchildpostlocation}{.}

```

\glslistdesc

```

16889 \newcommand{\glslistdesc}[1]{\glossentrydesc{#1}\glspostdescription}

```

glslistgroupskip

```

16890 \newcommand{\glslistgroupskip}{\nobreak\indexspace\nobreak}

```

\glslistitem

```

16891 \newcommand{\glslistitem}[1]{%
16892   \item[\glsentryitem{#1}%
16893     \glstarget{#1}{\glossentryname{#1}}]%
16894 }

```

\glslistinit This command was only added to glossary-list v4.48 so provide it if it hasn't been defined:

```

16895 \providecommand{\glslistinit}{%
16896   \ifdef\GetTitleStringDisableCommands
16897   {%
16898     \GetTitleStringSetup{expand}%
16899     \GetTitleStringDisableCommands{%
16900       \let\glsentryitem\@gobble
16901       \let\glstarget\@secondoftwo
16902       \let\glossentryname\glslistexpandedname
16903       \let\glslistgroupheaderfmt\@firstofone
16904       \let\glsgetgrouptitle\@firstofone

```

Technically this has an optional argument but it's not used in the list styles.

```

16905      \let\glsnavhypertarget\@secondoftwo
16906      \let\glsnavigation\relax
16907    }%
16908  }%
16909  {}%
16910 }

```

`istexpandedname` This command was only added to glossary-list v4.48 so provide it if it hasn't been defined. The original definition uses `\glsunexpandedfieldvalue` which was added to glossaries v4.48 (so if `\glslistexpandedname` hasn't been defined then neither will `\glsunexpandedfieldvalue`).

```

16911 \providecommand{\glslistexpandedname}[1]{%
16912   \ifcsname glo@\glsdetoklabel{#1}@name\endcsname
16913     \expandafter\expandonce\csname glo@\glsdetoklabel{#1}@name\expandafter\endcsname
16914   \fi
16915 }

```

Redefine list to use these commands.

```

16916 \renewglossarystyle{list}{%
16917   \renewenvironment{theglossary}%
16918     {\glslistinit\begin{description}}{\end{description}}%
16919   \renewcommand*\glossaryheader{}%
16920   \renewcommand*\glsgroupheading[1]{}%
16921   \renewcommand*\glossentry[2]{%
16922     \glslistitem{##1}\glslistdesc{##1}\glslistprelocation ##2}%
16923   \renewcommand*\subglossentry[3]{%
16924     \glssubentryitem{##2}%
16925     \glstarget{##2}{\strut}\space
16926     \glslistdesc{##2}%
16927     \glslistchildprelocation ##3\glslistchildpostlocation}%
16928   \renewcommand*\glsgroupskip{\ifglsnogroupskip\else\glslistgroupskip\fi}%
16929 }
16930 }
16931 {}

```

Similarly for `altlist`. Since it requires list, the new commands should have been defined above.

```

16932 \ifdef{\@glsstyle@altlist}
16933 {%

```

`\glsaltlistitem`

```

16934   \newcommand{\glsaltlistitem}[1]{%
16935     \glslistitem{#1}%
16936     \mbox{}\par\nobreak\@afterheading
16937   }

16938   \renewglossarystyle{altlist}{%
16939     \setglossarystyle{list}%
16940     \renewcommand*\glossentry[2]{%
16941       \glsaltlistitem{##1}%

```

```

16942     \glslistdesc{##1}\glslistprelocation ##2}%
16943     \renewcommand{\subglossentry}[3]{%
16944         \par
16945         \glssubentryitem{##2}%
16946         \glstarget{##2}{\strut}\glslistdesc{##2}%
16947         \glslistchildprelocation ##3}%
16948     }
16949 }
16950 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

16951 \ifdef{\@glsstyle@listgroup}
16952 {%

```

roupafterheader

```

16953     \newcommand{\glslistgroupheaderitem}[2]{\item[{#2}]}
16954     \newcommand{\glslistgroupafterheader}{%
16955         \mbox{}\par\nobreak\@afterheading
16956     }

16957     \renewglossarystyle{listgroup}{%
16958         \setglossarystyle{list}%
16959         \renewcommand*\{\glsgroupheading}[1]{%
16960             \glslistgroupheaderitem{##1}{\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
16961             \glslistgroupafterheader
16962         }%
16963     }
16964 }
16965 {}

```

Similarly for listhypergroup.

```

16966 \ifdef{\@glsstyle@listhypergroup}
16967 {%
16968     \renewglossarystyle{listhypergroup}{%
16969         \setglossarystyle{list}%
16970         \renewcommand*\{\glossaryheader}{%
16971             \glslistnavigationitem{\glsnavigation}}%
16972         \renewcommand*\{\glsgroupheading}[1]{%
16973             \glslistgroupheaderitem{##1}{\glslistgroupheaderfmt
16974                 {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}}%
16975             \glslistgroupafterheader
16976         }%
16977     }
16978 }
16979 {}

```

Similarly for altlistgroup.

```

16980 \ifdef{\@glsstyle@altlistgroup}
16981 {%
16982     \renewglossarystyle{altlistgroup}{%

```

```

16983 \setglossarystyle{altlist}%
16984 \renewcommand*{\glsgroupheading}[1]{%
16985 \glslistgroupheaderitem{##1}%
16986 {\glslistgroupheaderfmt{\glsgrouptitle{##1}}}%
16987 \glslistgroupafterheader
16988 }%
16989 }
16990 }
16991 {}

```

Similarly for altlisthypergroup.

```

16992 \ifdef{\@glsstyle@altlisthypergroup}
16993 {%
16994 \renewglossarystyle{altlisthypergroup}{%
16995 \setglossarystyle{altlist}%
16996 \renewcommand*{\glossaryheader}{%
16997 \glslistnavigationitem{\glsnavigation}}%
16998 \renewcommand*{\glsgroupheading}[1]{%
16999 \glslistgroupheaderitem{##1}{\glslistgroupheaderfmt
17000 {\glslnavhypertarget{##1}{\glsgrouptitle{##1}}}%
17001 \glslistgroupafterheader
17002 }%
17003 }
17004 }
17005 {}

```

2.3 Longtable Styles

The three and four column styles require adjustment to add the post-description hook. The two column styles need the hard-coded `\space` changed to `\glstrprelocation`.

```

17006 \ifcsdef{@glsstyle@long}
17007 {%
17008 \renewglossarystyle{long}{%
17009 \renewenvironment{theglossary}%
17010 {\begin{longtable}{lp{\glsgdescwidth}}}%
17011 {\end{longtable}}%
17012 \renewcommand*{\glossaryheader}{}%
17013 \renewcommand*{\glsgroupheading}[1]{}%
17014 \renewcommand{\glossentry}[2]{%
17015 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17016 \glossentrydesc{##1}\glspostdescription
17017 \glstrprelocation ##2\tabularnewline
17018 }%
17019 \renewcommand{\subglossentry}[3]{%
17020 &
17021 \glssubentryitem{##2}%
17022 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
17023 \glstrprelocation ##3\tabularnewline

```

```

17024 }%
17025 \ifglsnogroupskip
17026 \renewcommand*{\glsgroupskip}{}%
17027 \else
17028 \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
17029 \fi
17030 }
17031 }
17032 {}

```

Three column style:

```

17033 \ifcsdef{@glstyle@long3col}
17034 {%
17035 \renewglossarystyle{long3col}{%
17036 \renewenvironment{theglossary}%
17037 {\begin{longtable}\lp{\glstdescwidth}p{\glspagelistwidth}}}%
17038 {\end{longtable}}}%
17039 \renewcommand*{\glossaryheader}{}%
17040 \renewcommand*{\glsgroupheading}[1]{}%
17041 \renewcommand{\glossentry}[2]{%
17042 \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17043 \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
17044 }%
17045 \renewcommand{\subglossentry}[3]{%
17046 &
17047 \glssubentryitem{##2}%
17048 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
17049 ##3\tabularnewline
17050 }%

```

Conditional needs to be outside of `\glsgroupskip` otherwise it can cause “Incomplete `\iftrue`” errors.

```

17051 \ifglsnogroupskip
17052 \renewcommand*{\glsgroupskip}{}%
17053 \else
17054 \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
17055 \fi
17056 }
17057 }
17058 {}

```

Four column style:

```

17059 \ifcsdef{@glstyle@long4col}
17060 {%
17061 \renewglossarystyle{long4col}{%
17062 \renewenvironment{theglossary}%
17063 {\begin{longtable}{llll}}}%
17064 {\end{longtable}}}%
17065 \renewcommand*{\glossaryheader}{}%
17066 \renewcommand*{\glsgroupheading}[1]{}%
17067 \renewcommand{\glossentry}[2]{%

```

```

17068     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17069     \glossentrydesc{##1}\glspostdescription &
17070     \glossentrysymbol{##1} &
17071     ##2\tabularnewline
17072 }%
17073 \renewcommand{\subglossentry}[3]{%
17074     &
17075     \glssubentryitem{##2}%
17076     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
17077     \glossentrysymbol{##2} & ##3\tabularnewline
17078 }%

17079 \ifglsnogroupskip
17080     \renewcommand*\{glsgroupskip}{}%
17081 \else
17082     \renewcommand*\{glsgroupskip}{& & \tabularnewline}%
17083 \fi
17084 }
17085 }
17086 {}

```

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 for the post-description hook, but not the two column styles. However, the two-column styles need to have `\space` replaced with `\glxtrprelocation`.

```

17087 \ifcsdef{@glsstyle@longragged}
17088 {%
17089     \renewglossarystyle{longragged}{%
17090         \renewenvironment{theglossary}%
17091             {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}}}%
17092             {\end{longtable}}%
17093         \renewcommand*\{glossaryheader}{}%
17094         \renewcommand*\{glsgroupheading}[1]{}%
17095         \renewcommand{\glossentry}[2]{%
17096             \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17097             \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
17098             \tabularnewline
17099         }%
17100         \renewcommand{\subglossentry}[3]{%
17101             &
17102             \glssubentryitem{##2}%
17103             \glstarget{##2}{\strut}\glossentrydesc{##2}%
17104             \glspostdescription\glxtrprelocation ##3%
17105             \tabularnewline
17106         }%

```

```

17107 \ifglsnogroupskip
17108 \renewcommand*{\glsgroupskip}{}%
17109 \else
17110 \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
17111 \fi
17112 }
17113 }
17114 {}

```

Three and four column styles don't use `\glxtrprelocation` since the number list is in its own column.

```

17115 \ifcsdef{@glsstyle@longragged3col}
17116 {%
17117 \renewglossarystyle{longragged3col}{%
17118 \renewenvironment{theglossary}%
17119 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
17120 >{\raggedright}p{\glspagelistwidth}}}%
17121 {\end{longtable}}}%
17122 \renewcommand*{\glossaryheader}{}%
17123 \renewcommand*{\glsgroupheading}[1]{}%
17124 \renewcommand{\glossentry}[2]{%
17125 \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17126 \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
17127 }%
17128 \renewcommand{\subglossentry}[3]{%
17129 &
17130 \glssubentryitem{##2}%
17131 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
17132 ##3\tabularnewline
17133 }%
17134 \ifglsnogroupskip
17135 \renewcommand*{\glsgroupskip}{}%
17136 \else
17137 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
17138 \fi
17139 }
17140 }
17141 {}

```

Four column style:

```

17142 \ifcsdef{@glsstyle@altlongragged4col}
17143 {%
17144 \renewglossarystyle{altlongragged4col}{%
17145 \renewenvironment{theglossary}%
17146 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
17147 >{\raggedright}p{\glspagelistwidth}}}%
17148 {\end{longtable}}}%
17149 \renewcommand*{\glossaryheader}{}%
17150 \renewcommand*{\glsgroupheading}[1]{}%

```



```

17151 \renewcommand{\glossentry}[2]{%
17152   \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17153   \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
17154   ##2\tabularnewline
17155 }%
17156 \renewcommand{\subglossentry}[3]{%
17157   &
17158   \glssubentryitem{##2}%
17159   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
17160   \glossentrysymbol{##2} & ##3\tabularnewline
17161 }%

17162 \ifglsnogroupskip
17163   \renewcommand*\{\glsgroupskip}{}%
17164 \else
17165   \renewcommand*\{\glsgroupskip}{& & \tabularnewline}%
17166 \fi
17167 }
17168 }
17169 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment to add the post-description hook. The two column styles need the hard-coded `\space` changed to `\glxtrprelocation`.

```

17170 \ifcsdef{@glstyle@super}
17171 {%
17172   \renewglossarystyle{super}{%
17173     \renewenvironment{theglossary}%
17174       {\tablehead{}\tabletail{}}%
17175       \begin{supertabular}{\lp{\glsdescwidth}}%
17176       {\end{supertabular}}%
17177   \renewcommand*\{\glossaryheader}{}%
17178   \renewcommand*\{\glsgroupheading}[1]{}%
17179   \renewcommand{\glossentry}[2]{%
17180     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17181     \glossentrydesc{##1}\glspostdescription
17182     \glxtrprelocation ##2\tabularnewline
17183   }%
17184   \renewcommand{\subglossentry}[3]{%
17185     &
17186     \glssubentryitem{##2}%
17187     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
17188     \glxtrprelocation ##3\tabularnewline
17189   }%
17190   \ifglsnogroupskip
17191     \renewcommand*\{\glsgroupskip}{}%
17192   \else

```

```

17193     \renewcommand*{\glsgroupskip}{& \tabularnewline}%
17194 \fi
17195 }
17196 }
17197 {}

```

Three column style:

```

17198 \ifcsdef{@glsstyle@super3col}
17199 {%
17200   \renewglossarystyle{super3col}{%
17201     \renewenvironment{theglossary}%
17202       {\tablehead{}\tabletail{}}%
17203       \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
17204       {\end{supertabular}}%
17205     \renewcommand*{\glossaryheader}{}%
17206     \renewcommand*{\glsgroupheading}[1]{}%
17207     \renewcommand{\glossentry}[2]{%
17208       \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17209       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
17210     }%
17211     \renewcommand{\subglossentry}[3]{%
17212       &
17213       \glssubentryitem{##2}%
17214       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
17215       ##3\tabularnewline
17216     }%
17217   \ifglsgnogroupskip
17218     \renewcommand*{\glsgroupskip}{}%
17219   \else
17220     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
17221   \fi
17222 }
17223 }
17224 {}

```

Four column styles:

```

17225 \ifcsdef{@glsstyle@super4col}
17226 {%
17227   \renewglossarystyle{super4col}{%
17228     \renewenvironment{theglossary}%
17229       {\tablehead{}\tabletail{}}%
17230       \begin{supertabular}{llll}}{%
17231       \end{supertabular}}%
17232     \renewcommand*{\glossaryheader}{}%
17233     \renewcommand*{\glsgroupheading}[1]{}%
17234     \renewcommand{\glossentry}[2]{%
17235       \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17236       \glossentrydesc{##1}\glspostdescription &
17237       \glossentrysymbol{##1} & ##2\tabularnewline

```

```

17238 }%
17239 \renewcommand{\subglossentry}[3]{%
17240     &
17241     \glssubentryitem{##2}%
17242     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
17243     \glossentrysymbol{##2} & ##3\tabularnewline
17244 }%

17245 \ifglsnogroupskip
17246     \renewcommand*{\glsgroupskip}{}%
17247 \else
17248     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
17249 \fi
17250 }
17251 }
17252 {}

```

2.6 Super Ragged Styles

The three and four column styles require adjustment for the post-description hook, but not the two column styles. However, the two-column styles need to have `\space` replaced with `\glxstrprelocation`.

```

17253 \ifcsdef{@glstyle@superragged}
17254 {%
17255     \renewglossarystyle{superragged}{%
17256         \renewenvironment{theglossary}%
17257             {\tablehead{}\tabletail}%
17258             \begin{supertabular}{l>{\raggedright}p{\glstdescwidth}}%
17259             {\end{supertabular}}%
17260         \renewcommand*{\glossaryheader}{}%
17261         \renewcommand*{\glsgroupheading}[1]{}%
17262         \renewcommand{\glossentry}[2]{%
17263             \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17264             \glossentrydesc{##1}\glspostdescription\glxstrprelocation ##2%
17265             \tabularnewline
17266         }%
17267         \renewcommand{\subglossentry}[3]{%
17268             &
17269             \glssubentryitem{##2}%
17270             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
17271             \glxstrprelocation ##3%
17272             \tabularnewline
17273         }%
17274         \ifglsnogroupskip
17275             \renewcommand*{\glsgroupskip}{}%
17276         \else
17277             \renewcommand*{\glsgroupskip}{& \tabularnewline}%
17278         \fi

```

```

17279 }
17280 }
17281 {}

```

Three column style:

```

17282 \ifcsdef{@glsstyle@superragged3col}
17283 {%
17284   \renewglossarystyle{superragged3col}{%
17285     \renewenvironment{theglossary}%
17286       {\tablehead{}\tabletail{}}%
17287       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%
17288         >{\raggedright}p{\glspagelistwidth}}}%
17289       {\end{supertabular}}%
17290     \renewcommand*{\glossaryheader}{}%
17291     \renewcommand*{\glsgroupheading}[1]{}%
17292     \renewcommand{\glossentry}[2]{%
17293       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17294       \glossentrydesc{##1}\glspostdescription &
17295       ##2\tabularnewline
17296     }%
17297     \renewcommand{\subglossentry}[3]{%
17298       &
17299       \glssubentryitem{##2}%
17300       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
17301       ##3\tabularnewline
17302     }%
17303   \ifglsnogroupskip
17304     \renewcommand*{\glsgroupskip}{}%
17305   \else
17306     \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
17307   \fi
17308 }
17309 }
17310 {}

```

Four columns:

```

17311 \ifcsdef{@glsstyle@altsuperragged4col}
17312 {%
17313   \renewglossarystyle{altsuperragged4col}{%
17314     \renewenvironment{theglossary}%
17315       {\tablehead{}\tabletail{}}%
17316       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}1%
17317         >{\raggedright}p{\glspagelistwidth}}}%
17318       {\end{supertabular}}%
17319     \renewcommand*{\glossaryheader}{}%
17320     \renewcommand{\glossentry}[2]{%
17321       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
17322       \glossentrydesc{##1}\glspostdescription &
17323       \glossentrysymbol{##1} & ##2\tabularnewline

```

```

17324 }%
17325 \renewcommand{\subglossentry}[3]{%
17326     &
17327     \glssubentryitem{##2}%
17328     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
17329     \glossentrysymbol{##2} & ##3\tabularnewline
17330 }%

17331 \ifglsnogroupskip
17332     \renewcommand*{\glsgroupskip}{}%
17333 \else
17334     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
17335 \fi
17336 }
17337 }
17338 {}

```

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`.

```

17339 \ifdef{\@glsstyle@inline}
17340 {%
17341     \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}

    Just use \glxtrpostdescription instead of \glspostdescription.

17342     \renewcommand*{\glsinlinedescformat}[3]{%
17343         \space#1\glxtrpostdescription}
17344     \renewcommand*{\glsinlinesubdescformat}[3]{%
17345         #1\glxtrpostdescription}

```

The default settings don't show the location lists, so there's no adjustment for `\glxtrprelocation`.

```

17346 }
17347 {}

```

2.8 Tree Styles

Redefine both `\glstreenamefmt` and `\glstreegroupheaderfmt` in terms of `\glstreedefaultnamefmt` to make it easier to change both at the same time or only change one without affecting the other.

```

17348 \ifdef\glstreenamefmt
17349 {
\glstreedefaultnamefmt
17350     \newcommand{\glstreedefaultnamefmt}[1]{\textbf{#1}}

```

`\glstreenamefmt`

```
17351 \renewcommand{\glstreenamefmt}[1]{\glstreedefaultnamefmt{#1}}
```

`\glstreegroupheaderfmt` This command was only introduced to glossary-tree v4.22, so it may not be defined.

```
17352 \def\glstreegroupheaderfmt#1{\glstreedefaultnamefmt{#1}}
```

`\glstreenavigationfmt` This command was only introduced to glossary-tree v4.22, so it may not be defined.

```
17353 \def\glstreenavigationfmt#1{\glstreedefaultnamefmt{#1}}
```

`\glstreePreHeader` Takes the label as the first argument and title as the second argument so this can be modified to add a bookmark.

```
17354 \newcommand{\glstreePreHeader}[2]{}
```

```
17355 }
```

```
17356 {}
```

The index style is redefined so that the space before the number list isn't hard coded.

```
17357 \ifdef{\@glsstyle@index}
```

```
17358 {
```

`\glstreeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```
17359 \newcommand*{\glstreeprelocation}{\glstxtrprelocation}
```

`\glstreechildprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```
17360 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}
```

Don't prohibit a page break at the start of a new group if there's no header.

`\glstreegroupskip`

```
17361 \newcommand{\glstreegroupskip}{\indexspace}
```

`\glstreegroupheaderskip` This doesn't include `\@afterheading` as it can cause interference with some styles.

```
17362 \newcommand{\glstreegroupheaderskip}{\nopagebreak\glstreegroupskip\nobreak}
```

Modify the index style.

```
17363 \renewglossarystyle{index}{%
17364   \renewenvironment{theglossary}%
17365     {\setlength{\parindent}{0pt}%
17366      \setlength{\parskip}{0pt plus 0.3pt}%
17367      \let\item\glstreeitem
17368      \let\subitem\glstreesubitem
17369      \let\subsubitem\glstreesubsubitem
17370     }%
17371   {\par}%
17372   \renewcommand*{\glossaryheader}{}%
17373   \renewcommand*{\glsgroupheading}[1]{}%
17374   \renewcommand*{\glossentry}[2]{%
17375     \item\glsentryitem{#1}%
```

```

17376     \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
17377     \glstreesymbol{##1}%
17378     \glstreeDescLoc{##1}{##2}%
17379 }%
17380 \renewcommand{\subglossentry}[3]{%
17381     \ifcase##1\relax
17382     \item
17383     \or
17384     \subitem
17385     \glssubentryitem{##2}%
17386     \else
17387     \subsubitem
17388     \fi
17389     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
17390     \glstreechildsymbol{##2}%
17391     \glstreeChildDescLoc{##2}{##3}%
17392 }%
17393 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\glstreegroupskip\fi}%
17394 }
17395 }
17396 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

17397 \ifdef{\@glsstyle@indexgroup}
17398 {%
17399     \renewglossarystyle{indexgroup}{%
17400         \setglossarystyle{index}%
17401         \renewcommand*{\glsgroupheading}[1]{%
17402             \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
17403             \glstreePreHeader{##1}{\glsxtr@grptitle}%
17404             \item\glstreegroupheaderfmt{\glsxtr@grptitle}%
17405             \glstreegroupheaderskip\@afterheading
17406         }%
17407     }
17408 }
17409 {}

```

Similarly for indexhypergroup.

```

17410 \ifdef{\@glsstyle@indexhypergroup}
17411 {%
17412     \renewglossarystyle{indexhypergroup}{%
17413         \setglossarystyle{index}%
17414         \renewcommand*{\glossaryheader}{%
17415             \item\glstreenavigationfmt{\glsnavigation}%
17416             \glstreegroupheaderskip\@afterheading}%
17417         \renewcommand*{\glsgroupheading}[1]{%
17418             \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
17419             \glstreePreHeader{##1}{\glsxtr@grptitle}%
17420             \item\glstreegroupheaderfmt
17421                 {\glsnahypertarget{##1}{\glsxtr@grptitle}}%

```

```

17422 \glstreegroupheaderskip\@afterheading}%
17423 }%
17424 }
17425 {}

```

Adjust tree style to remove hard coded space before number list.

```

17426 \ifdef{\@glsstyle@tree}
17427 {%

```

The original altree style doesn't use \glstreepredesc but since v1.42 the modified style (below) has switched to using \glstreeDescLoc so provide an alternative that can be used with altree.

sxtrtreepredesc

```

17428 \newcommand{\glsxtrtreepredesc}{\glstreepredesc}

```

reechildpredesc

```

17429 \newcommand{\glsxtrtreechildpredesc}{\glstreechildpredesc}

```

Provide a command for use with the tree styles that displays the pre-description separator, the description and post-description hook.

\glstreedesc

```

17430 \newcommand{\glstreedesc}[1]{%
17431 \glsxtrtreepredesc\glossentrydesc{#1}\glspostdescription
17432 }

```

\glstreeDescLoc

```
\glstreeDescLoc{<label>}{<location>}
```

This checks for the description and symbol. If both are missing, a different separator may be required. For example, a comma and space if there's no description or symbol but just a space if either of those fields are present.

```

17433 \newcommand{\glstreeDescLoc}[2]{%
17434 \ifglshasdesc{#1}%
17435 {\glstreedesc{#1}\glstreeprelocation}%
17436 {\ifglshassymbol{#1}{\glstreeprelocation}{\glstreeNoDescSymbolPreLocation}}}%
17437 #2%
17438 }

```

mbolPreLocation

```
\glstreeNoDescSymbolPreLocation
```

```

17439 \newcommand{\glstreeNoDescSymbolPreLocation}{\space}

```


Similarly for the symbol.

`\glstreesymbol`

```
17440 \newcommand{\glstreesymbol}[1]{%
17441   \ifglshassymbol{#1}{\space(\glossentrysymbol{#1})}{}%
17442 }%
```

And for the child entries:

`lstreechilddesc`

```
17443 \newcommand{\glstreechilddesc}[1]{%
17444   \glxtrtreechildpredesc\glossentrydesc{#1}\glspostdescription
17445 }%
```

`reeChildDescLoc`

```
17446 \newcommand{\glstreeChildDescLoc}[2]{%
17447   \ifglshasdesc{#1}%
17448   {\glstreechilddesc{#1}\glstreechildprelocation}%
17449   {\ifglshassymbol{#1}{\glstreechildprelocation}%
17450    {\glstreeNoDescSymbolPreLocation}%
17451   }%
17452   #2%
17453 }%
```

`treechildsymbol` This just behaves in the same way as the top-level.

```
17454 \newcommand{\glstreechildsymbol}[1]{%
17455   \glstreesymbol{#1}%
17456 }%

17457 \renewglossarystyle{tree}{%
17458   \renewenvironment{theglossary}%
17459     {\setlength{\parindent}{0pt}%
17460      \setlength{\parskip}{0pt plus 0.3pt}}%
17461   {}%
17462   \renewcommand*{\glossaryheader}{}%
17463   \renewcommand*{\glsgroupheading}[1]{}%
17464   \renewcommand{\glossentry}[2]{%
17465     \hangindent0pt\relax
17466     \parindent0pt\relax
17467     \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
17468     \glstreesymbol{##1}%
17469     \glstreeDescLoc{##1}{##2}\par
17470   }%
17471   \renewcommand{\subglossentry}[3]{%
17472     \hangindent##1\glstreeindent\relax
17473     \parindent##1\glstreeindent\relax
17474     \ifnum##1=1\relax
17475       \glssubentryitem{##2}%
17476     \fi
17477     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%

```

```

17478     \glstreechildsymbol{##2}%
17479     \glstreeChildDescLoc{##2}{##3}\par
17480   }%
17481   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\glstreegroupskip\fi}%
17482 }%
17483 }
17484 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

17485 \ifdef{\@glsstyle@treegroup}
17486 {%
17487   \renewglossarystyle{treegroup}{%
17488     \setglossarystyle{tree}%
17489     \renewcommand{\glsgroupheading}[1]{%
17490       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
17491       \glstreePreHeader{##1}{\glstr@grptitle}%
17492       \par\noindent\glstreegroupheaderfmt{\glstr@grptitle}%
17493       \glstreegroupheaderskip\@afterheading}%
17494   }
17495 }
17496 {}

```

Similarly for treehypergroup

```

17497 \ifdef{\@glsstyle@treehypergroup}
17498 {%
17499   \renewglossarystyle{treehypergroup}{%
17500     \setglossarystyle{tree}%
17501     \renewcommand*{\glossaryheader}{%
17502       \par\noindent\glstreenavigationfmt{\glsnavigation}%
17503       \glstreegroupheaderskip\@afterheading}%
17504     \renewcommand*{\glsgroupheading}[1]{%
17505       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
17506       \glstreePreHeader{##1}{\glstr@grptitle}%
17507       \par\noindent
17508       \glstreegroupheaderfmt
17509         {\glsnavhypertarget{##1}{\glstr@grptitle}}%
17510       \glstreegroupheaderskip\@afterheading}%
17511   }
17512 }
17513 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

17514 \ifdef{\@glsstyle@treenoname}
17515 {%

```

Provide a command for use with the treenoname styles that displays the pre-description separator, the description and post-description hook.

streenonamedesc

```

17516   \newcommand{\glstreenonamedesc}[1]{%
17517     \glstreepredesc\glossentrydesc{#1}\glspostdescription

```

17518 }%

Similarly for the symbol.

treenonamesymbol

```
17519 \newcommand{\glstreenonamesymbol}[1]{%
17520   \ifglshassymbol{#1}{\space(\glossentrysymbol{#1})}{}%
17521 }%
```

treenonameDescLoc

```
17522 \newcommand{\glstreenonameDescLoc}[2]{%
17523   \glstreenonamedesc{#1}\glstreeprelocation#2%
17524 }
```

nonamechilddesc The child entry doesn't have the pre-description separator as the name isn't displayed.

```
17525 \newcommand{\glstreenonamechilddesc}[1]{%
17526   \glossentrydesc{#1}\glspostdescription
17527 }%
```

nameChildDescLoc

```
17528 \newcommand{\glstreenonameChildDescLoc}[2]{%
17529   \glstreenonamechilddesc{#1}\glstreechildprelocation#2%
17530 }

17531 \renewglossarystyle{treenoname}{%
17532   \renewenvironment{theglossary}%
17533     {\setlength{\parindent}{0pt}%
17534      \setlength{\parskip}{0pt plus 0.3pt}}%
17535   {%
17536     \renewcommand*{\glossaryheader}{}%
17537     \renewcommand*{\glsgroupheading}[1]{}%
17538     \renewcommand{\glossentry}[2]{%
17539       \hangindent0pt\relax
17540       \parindent0pt\relax
17541       \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
17542       \glstreenamesymbol{##1}%

17543       \glstreenonameDescLoc{##1}{##2}\par
17544     }%
17545     \renewcommand{\subglossentry}[3]{%
17546       \hangindent##1\glstreeindent\relax
17547       \parindent##1\glstreeindent\relax
17548       \ifnum##1=1\relax
17549         \glssubentryitem{##2}%
17550       \fi
17551       \glstarget{##2}{\strut}%
17552       \glstreenonameChildDescLoc{##2}{##3}\par
17553     }%
17554     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\glstreegroupskip\fi}%
17555   }
```

```
17556 }
17557 {}
```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```
17558 \ifdef{\@glsstyle@treenonamegroup}
17559 {%
17560   \renewglossarystyle{treenonamegroup}{%
17561     \setglossarystyle{treenoname}%
17562     \renewcommand{\glsgroupheading}[1]{%
17563       \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
17564       \glstreePreHeader{##1}{\glsxtr@grptitle}%
17565       \par\noindent\glstreegroupheaderfmt{\glsxtr@grptitle}%
17566       \glstreegroupheaderskip\@afterheading
17567     }%
17568   }
17569 }
17570 {}
```

Similarly for `treenonamehypergroup`

```
17571 \ifdef{\@glsstyle@treenonamehypergroup}
17572 {%
17573   \renewglossarystyle{treenonamehypergroup}{%
17574     \setglossarystyle{treenoname}%
17575     \renewcommand*\{\glossaryheader}{%
17576       \par\noindent\glstreenavigationfmt{\glsnavigation}%
17577       \glstreegroupheaderskip\@afterheading}%
17578     \renewcommand*\{\glsgroupheading}[1]{%
17579       \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
17580       \glstreePreHeader{##1}{\glsxtr@grptitle}%
17581       \par\noindent
17582       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsxtr@grptitle}}}%
17583     \glstreegroupheaderskip\@afterheading}%
17584   }
17585 }
17586 {}
```

The `alttree` style is redefined to make it easier to made minor adjustments.

```
17587 \ifdef{\@glsstyle@alttree}
17588 {%
```

Only redefine this style if it's already been defined.

`salttreepredesc`

```
17589 \newcommand{\glsalttreepredesc}{}

```

`reechildpredesc`

```
17590 \newcommand{\glsalttreechildpredesc}{\glsalttreepredesc}

```

`bolDescLocation`

`\glxtralttreeSymbolDescLocation{<label>}{<location list>}`

Layout the symbol, description and location for top-level entries.

```

17591 \newcommand{\glxtralttreeSymbolDescLocation}[2]{%
17592   {%
17593     \let\par\glxtrAltTreePar

17594     \let\glxtrtreepredesc\glsaltpredecessor
17595     \let\glxtrtreechildpredesc\glsaltpredecessor
17596     \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%

17597     \glstreeDescLoc{#1}{#2}\par
17598   }%
17599 }

```

`trAltTreeIndent` Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

17600 \newlength\glxtrAltTreeIndent

```

`lsxtrAltTreePar` Multi-paragraph descriptions need to keep the hanging indent.

```

17601 \newcommand{\glxtrAltTreePar}{%
17602   \@par
17603   \glxtrAltTreeSetHangIndent
17604   \setlength{\parindent}{\dimexpr\hangindent+\glxtrAltTreeIndent}%
17605 }

```

`bolDescLocation`

`\glxtralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

17606 \newcommand{\glxtralttreeSubSymbolDescLocation}[3]{%
17607   \glxtralttreeSymbolDescLocation{#2}{#3}%
17608 }

```

`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.

```

17609 \newlength\glxtrtreetopindent

```

`lsxtralttreeInit` User-level initialisation for the alttree style.

```

17610 \newcommand*{\glxtralttreeInit}{%
17611   \settowidth{\glxtrtreetopindent}{\glstreenamfmt{\glsgetwidestname\space}}%
17612   \glxtrAltTreeIndent=\parindent
17613 }

```

`\glssetwidest` The original `\glssetwidest` only uses `\def`. This uses `\gdef`.

```

17614 \newcommand*{\glssetwidest}[2][0]{%
17615   \csgdef{@glswidestname\romannumeral#1}{#2}%
17616 }

```

`\eglssetwidest` The original `\glssetwidest` only uses `\def`. This uses `\protected@csedef`.

```

17617 \newcommand*{\eglssetwidest}[2][0]{%
17618   \protected@csedef{@glswidestname\romannumeral#1}{#2}%
17619 }

```

`\xglssetwidest` Like the above but uses `\protected@csxdef`.

```

17620 \newcommand*{\xglssetwidest}[2][0]{%
17621   \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
17622 }

```

`glsupdatewidest` Only sets if new value is wider than old value.

```

17623 \newcommand*{\glsupdatewidest}[2][0]{%
17624   \ifcsundef{@glswidestname\romannumeral#1}%
17625   {\csdef{@glswidestname\romannumeral#1}{#2}}%
17626   {%
17627     \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
17628     \settowidth{\dimen@ii}{#2}%
17629     \ifdim\dimen@ii>\dimen@
17630     \csdef{@glswidestname\romannumeral#1}{#2}%
17631     \fi
17632   }%
17633 }

```

`glsupdatewidest` As above but global definition.

```

17634 \newcommand*{\glsupdatewidest}[2][0]{%
17635   \ifcsundef{@glswidestname\romannumeral#1}%
17636   {\csgdef{@glswidestname\romannumeral#1}{#2}}%
17637   {%
17638     \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
17639     \settowidth{\dimen@ii}{#2}%
17640     \ifdim\dimen@ii>\dimen@
17641     \csgdef{@glswidestname\romannumeral#1}{#2}%
17642     \fi
17643   }%
17644 }

```

`glsupdatewidest` As `\glsupdatewidest` but expands value.

```

17645 \newcommand*{\eglsupdatewidest}[2][0]{%
17646   \ifcsundef{@glswidestname\romannumeral#1}%
17647   {\protected@csedef{@glswidestname\romannumeral#1}{#2}}%
17648   {%
17649     \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
17650     \settowidth{\dimen@ii}{#2}%

```

```

17651      \ifdim\dimen@ii>\dimen@
17652      \protected@csedef{@glswidestname\romannumeral#1}{#2}%
17653      \fi
17654  }%
17655  }

```

glsupdatewidest As above but global.

```

17656  \newcommand*{\xglsupdatewidest}[2][0]{%
17657    \ifcsundef{@glswidestname\romannumeral#1}%
17658    {\protected@csxdef{@glswidestname\romannumeral#1}{#2}}%
17659    {%
17660      \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
17661      \settowidth{\dimen@ii}{#2}%
17662      \ifdim\dimen@ii>\dimen@
17663      \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
17664      \fi
17665    }%
17666  }

```

glswidestname Provide a user-level macro to obtain the widest top-level name.

```

17667  \newcommand*{\glswidestname}{\@glswidestname}

```

glswidestsubname Provide a user-level macro to obtain the widest sub-entry name.

```

17668  \newcommand*{\glswidestsubname}[1]{%
17669    \ifcsundef{@glswidestname\romannumeral#1}%
17670    {\@glswidestname}%
17671    {\csuse{@glswidestname\romannumeral#1}}%
17672  }

```

glswidestTopLevelName CamelCase is easier for long command names. Provide a CamelCase synonym of `\glswidestTopLevelName`.

```

17673  \let\glswidestTopLevelName\glswidestTopLevelName

```

glswidestUsedTopLevelName Like `\glswidestTopLevelName` 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.

```

17674  \newrobustcmd*{\glswidestUsedTopLevelName}[1][\@glo@types]{%
17675    \dimen@=0pt\relax
17676    \glstmplen=0pt\relax
17677    \foralllglossaries[#1]{\@gls@type}%
17678    {%
17679      \forallgsentries[\@gls@type]{\@glo@label}%
17680      {%
17681        \ifglswidestUsed{\@glo@label}%
17682        {%
17683          \ifglshasparent{\@glo@label}%
17684          {}%
17685          {%

```

```

17686         \settowidth{\dimen@}%
17687         {\glstreenamefmt{\glstryname{\@glo@label}}}%
17688         \ifdim\dimen@>\gls@tmplen
17689             \gls@tmplen=\dimen@
17690             \eglssetwidest{\glstryname{\@glo@label}}%
17691         \fi
17692     }%
17693 }%
17694 {}%
17695 }%
17696 }%
17697 }

```

destUsedAnyName Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

17698 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
17699     \dimen@=0pt\relax
17700     \gls@tmplen=0pt\relax
17701     \foralllglossaries[#1]{\@gls@type}%
17702     {%
17703         \forglstries[\@gls@type]{\@glo@label}%
17704         {%
17705             \ifglused{\@glo@label}%
17706             {%
17707                 \settowidth{\dimen@}%
17708                 {\glstreenamefmt{\glstryname{\@glo@label}}}%
17709                 \ifdim\dimen@>\gls@tmplen
17710                     \gls@tmplen=\dimen@
17711                     \eglssetwidest{\glstryname{\@glo@label}}%
17712             \fi
17713         }%
17714     }%
17715 }%
17716 }%
17717 }

```

ndWidestAnyName Like the above but doesn't check is the entry has been used.

```

17718 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
17719     \dimen@=0pt\relax
17720     \gls@tmplen=0pt\relax
17721     \foralllglossaries[#1]{\@gls@type}%
17722     {%
17723         \forglstries[\@gls@type]{\@glo@label}%
17724         {%
17725             \settowidth{\dimen@}%
17726             {\glstreenamefmt{\glstryname{\@glo@label}}}%
17727             \ifdim\dimen@>\gls@tmplen
17728                 \gls@tmplen=\dimen@
17729                 \eglssetwidest{\glstryname{\@glo@label}}%

```



```

17730     \fi
17731   }%
17732 }%
17733 }

```

estUsedLevelTwo This is like \glsFindWidestUsedTopLevelName but also sets the first two sub-levels as well.

Any entry that has a great-grandparent is ignored.

```

17734 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
17735   \dimen@=0pt\relax
17736   \dimen@i=0pt\relax
17737   \dimen@ii=0pt\relax
17738   \forallglossaries[#1]{\@gls@type}%
17739   {%
17740     \forglsentries[\@gls@type]{\@glo@label}%
17741     {%
17742       \ifglsused{\@glo@label}%
17743       {%
17744         \ifglshasparent{\@glo@label}%
17745         {%
17746           \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
17747           \ifglshasparent{\@glo@parent}%
17748           {%
17749             \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
17750             \ifglshasparent{\@glo@parent}%
17751             {%
17752               {%
17753                 \settowidth{\gls@tmplen}%
17754                   {\glstreenamfmt{\glsentryname{\@glo@label}}}%
17755                 \ifdim\gls@tmplen>\dimen@ii
17756                   \dimen@ii=\gls@tmplen
17757                   \eglssetwidest[2]{\glsentryname{\@glo@label}}%
17758                 \fi
17759               }%
17760             }%
17761           {%
17762             \settowidth{\gls@tmplen}%
17763               {\glstreenamfmt{\glsentryname{\@glo@label}}}%
17764             \ifdim\gls@tmplen>\dimen@i
17765               \dimen@i=\gls@tmplen
17766               \eglssetwidest[1]{\glsentryname{\@glo@label}}%
17767             \fi
17768           }%
17769         }%
17770       {%
17771         \settowidth{\gls@tmplen}%
17772           {\glstreenamfmt{\glsentryname{\@glo@label}}}%
17773         \ifdim\gls@tmplen>\dimen@
17774           \dimen@=\gls@tmplen
17775           \eglssetwidest{\glsentryname{\@glo@label}}%

```

```

17776         \fi
17777     }%
17778 }%
17779 {}%
17780 }%
17781 }%
17782 }

```

dWidestLevelTwo This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

17783 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
17784     \dimen@=0pt\relax
17785     \dimen@i=0pt\relax
17786     \dimen@ii=0pt\relax
17787     \foralllglossaries[#1]{\@gls@type}%
17788     {%
17789         \forglsentries[\@gls@type]{\@glo@label}%
17790         {%
17791             \ifglshasparent{\@glo@label}%
17792             {%
17793                 \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
17794                 \ifglshasparent{\@glo@parent}%
17795                 {%
17796                     \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
17797                     \ifglshasparent{\@glo@parent}%
17798                     {}%
17799                     {%
17800                         \settowidth{\gls@tmplen}%
17801                             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
17802                         \ifdim\gls@tmplen>\dimen@ii
17803                             \dimen@ii=\gls@tmplen
17804                             \eglssetwidest[2]{\glsentryname{\@glo@label}}%
17805                         \fi
17806                     }%
17807                 }%
17808             }%
17809             \settowidth{\gls@tmplen}%
17810                 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
17811             \ifdim\gls@tmplen>\dimen@i
17812                 \dimen@i=\gls@tmplen
17813                 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
17814             \fi
17815         }%
17816     }%
17817     {%
17818         \settowidth{\gls@tmplen}%
17819             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
17820         \ifdim\gls@tmplen>\dimen@
17821             \dimen@=\gls@tmplen
17822             \eglssetwidest{\glsentryname{\@glo@label}}%

```

```

17823         \fi
17824     }%
17825 }%
17826 }%
17827 }

```

`\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.

```

17828 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
17829     \dimen@=0pt\relax
17830     \gls@tmplen=0pt\relax
17831     #2=0pt\relax
17832     \forallglossaries[#1]{\@gls@type}%
17833     {%
17834         \forglsentries[\@gls@type]{\@glo@label}%
17835         {%
17836             \ifglsused{\@glo@label}%
17837             {%
17838                 \settowidth{\dimen@}%
17839                 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
17840                 \ifdim\dimen@>\gls@tmplen
17841                     \gls@tmplen=\dimen@
17842                     \eglssetwidest{\glsentryname{\@glo@label}}%
17843                 \fi
17844                 \settowidth{\dimen@}%
17845                 {\glsentrysymbol{\@glo@label}}%
17846                 \ifdim\dimen@>#2\relax
17847                     #2=\dimen@
17848                 \fi
17849             }%
17850         }%
17851     }%
17852 }%
17853 }

```

`\stAnyNameSymbol` Like the above but doesn't check if the entry has been used.

```

17854 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
17855     \dimen@=0pt\relax
17856     \gls@tmplen=0pt\relax
17857     #2=0pt\relax
17858     \forallglossaries[#1]{\@gls@type}%
17859     {%
17860         \forglsentries[\@gls@type]{\@glo@label}%
17861         {%
17862             \settowidth{\dimen@}%
17863             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
17864             \ifdim\dimen@>\gls@tmplen
17865                 \gls@tmplen=\dimen@
17866                 \eglssetwidest{\glsentryname{\@glo@label}}%

```

```

17867         \fi
17868         \settowidth{\dimen@}%
17869         {\glsentrysymbol{\@glo@label}}}%
17870         \ifdim\dimen@>#2\relax
17871         #2=\dimen@
17872         \fi
17873     }%
17874 }%
17875 }

```

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.

```

17876 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
17877     \dimen@=0pt\relax
17878     \gls@tmplen=0pt\relax
17879     #2=0pt\relax
17880     #3=0pt\relax
17881     \foralllglossaries[#1]{\@gls@type}%
17882     {%
17883         \forglentries[\@gls@type]{\@glo@label}%
17884         {%
17885             \ifglused{\@glo@label}%
17886             {%
17887                 \settowidth{\dimen@}%
17888                 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
17889                 \ifdim\dimen@>\gls@tmplen
17890                 \gls@tmplen=\dimen@
17891                 \eglssetwidest{\glsentryname{\@glo@label}}%
17892                 \fi
17893                 \settowidth{\dimen@}%
17894                 {\glsentrysymbol{\@glo@label}}}%
17895                 \ifdim\dimen@>#2\relax
17896                 #2=\dimen@
17897                 \fi
17898                 \settowidth{\dimen@}%
17899                 {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
17900                 \ifdim\dimen@>#3\relax
17901                 #3=\dimen@
17902                 \fi
17903             }%
17904         }%
17905     }%
17906 }%
17907 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

17908 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%

```

```

17909 \dimen@=0pt\relax
17910 \gls@tmplen=0pt\relax
17911 #2=0pt\relax
17912 #3=0pt\relax
17913 \forall glossaries[#1]{\@gls@type}%
17914 {%
17915   \forall sentries[\@gls@type]{\@glo@label}%
17916   {%
17917     \settowidth{\dimen@}%
17918     {\glstreenamfmt{\glsentryname{\@glo@label}}}%
17919     \ifdim\dimen@>\gls@tmplen
17920       \gls@tmplen=\dimen@
17921       \eglssetwidest{\glsentryname{\@glo@label}}%
17922     \fi
17923     \settowidth{\dimen@}%
17924     {\glsentrysymbol{\@glo@label}}%
17925     \ifdim\dimen@>#2\relax
17926       #2=\dimen@
17927     \fi
17928     \settowidth{\dimen@}%
17929     {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
17930     \ifdim\dimen@>#3\relax
17931       #3=\dimen@
17932     \fi
17933   }%
17934 }%
17935 }

```

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.

```

17936 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
17937   \dimen@=0pt\relax
17938   \gls@tmplen=0pt\relax
17939   #2=0pt\relax
17940   \forall glossaries[#1]{\@gls@type}%
17941   {%
17942     \forall sentries[\@gls@type]{\@glo@label}%
17943     {%
17944       \ifglsused{\@glo@label}%
17945       {%
17946         \settowidth{\dimen@}%
17947         {\glstreenamfmt{\glsentryname{\@glo@label}}}%
17948         \ifdim\dimen@>\gls@tmplen
17949           \gls@tmplen=\dimen@
17950           \eglssetwidest{\glsentryname{\@glo@label}}%
17951         \fi
17952         \settowidth{\dimen@}%
17953         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%

```

```

17954         \ifdim\dimen@>#2\relax
17955             #2=\dimen@
17956         \fi
17957     }%
17958     {}%
17959 }%
17960 }%
17961 }

```

AnyNameLocation Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

17962 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
17963     \dimen@=0pt\relax
17964     \gls@tmplen=0pt\relax
17965     #2=0pt\relax
17966     \forallglossaries[#1]{\@gls@type}%
17967     {%
17968         \forglsentries[\@gls@type]{\@glo@label}%
17969         {%
17970             \settowidth{\dimen@}%
17971             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
17972             \ifdim\dimen@>\gls@tmplen
17973                 \gls@tmplen=\dimen@
17974                 \eglssetwidest{\glsentryname{\@glo@label}}%
17975             \fi
17976             \settowidth{\dimen@}%
17977             {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
17978             \ifdim\dimen@>#2\relax
17979                 #2=\dimen@
17980             \fi
17981         }%
17982     }%
17983 }

```

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`.

```

17984 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
17985     \glstreeindent=\glsxtrtreetopindent\relax
17986 }

```

teTreeSubIndent

`\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.

```

17987 \newcommand*{\glxtrComputeTreeSubIndent}[3]{%
17988   \ifcsundef{@glswidestname\romannumeral#1}%
17989   {%
17990     \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
17991   }%
17992   {%
17993     \settowidth{#3}{\glstreenamefmt{%
17994       \csname @glswidestname\romannumeral#1\endcsname\space}}%
17995   }%
17996 }

```

eeSetHangIndent Set \hangindent for top-level entries:

```

17997 \newcommand*{\glxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set \hangindent for sub-entries:

```

17998 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine alttree:

```

17999 \renewglossarystyle{alttree}{%
18000   \renewenvironment{theglossary}%
18001   {%
18002     \glxtralttreeInit
18003     \def\@gls@prevlevel{-1}%
18004     \mbox{}\par}%
18005   {\par}%
18006   \renewcommand*{\glossaryheader}{}%
18007   \renewcommand*{\glsgroupheading}[1]{}%
18008   \renewcommand{\glossentry}[2]{%
18009     \ifnum\@gls@prevlevel=0\relax
18010     \else
18011       \glxtrComputeTreeIndent{##1}%
18012     \fi
18013     \parindent\glstreeindent
18014     \glxtrAltTreeSetHangIndent
18015     \makebox[Opt][r]%
18016     {%
18017       \glstreenamebox{\glstreeindent}%
18018       {%
18019         \glsentryitem{##1}%
18020         \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
18021       }%
18022     }%
18023     \glxtralttreeSymbolDescLocation{##1}{##2}%
18024     \def\@gls@prevlevel{0}%
18025   }
18026   \renewcommand{\subglossentry}[3]{%
18027     \ifnum##1=1\relax
18028     \glssubentryitem{##2}%
18029   \fi

```

```

18030 \ifnum\@gls@prevlevel=##1\relax
18031 \else
18032 \glstrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
18033 \ifnum\@gls@prevlevel<##1\relax
18034 \setlength\glstreeindent\gls@tmplen
18035 \addtolength\glstreeindent\parindent
18036 \parindent\glstreeindent
18037 \else
18038 \ifnum\@gls@prevlevel=0\relax
18039 \glstrComputeTreeIndent{##2}%
18040 \else
18041 \glstrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
18042 \fi
18043 \addtolength\parindent{-\glstreeindent}%
18044 \setlength\glstreeindent\parindent
18045 \fi
18046 \fi
18047 \glstrAltTreeSetSubHangIndent{##1}%
18048 \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
18049 \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
18050 \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
18051 \def\@gls@prevlevel{##1}%
18052 }%
18053 \renewcommand*\@glsgroupskip{\ifglsnogroupskip\else\glstreegroupskip\fi}%
18054 }
18055 }%
18056 {%
18057 }

```

Redefine `almtreegroup` so that it discourages a break after group headings.

```

18058 \ifdef{\@glsstyle@almtreegroup}
18059 {%
18060 \renewglossarystyle{almtreegroup}{%
18061 \setglossarystyle{almtree}%
18062 \renewcommand{\glsgroupheading}[1]{\par
18063 \def\@gls@prevlevel{-1}%
18064 \hangindent0pt\relax
18065 \parindent0pt\relax
18066 \glstrgetgrouptitle{##1}{\glstr@grptitle}%
18067 \glstreePreHeader{##1}{\glstr@grptitle}%
18068 \glstreegroupheaderfmt{\glstr@grptitle}%

```

Can't use `\@afterheading` here as it messes with the first item of the group.

```

18069 \glstreegroupheaderskip
18070 }%
18071 }%
18072 }%
18073 {%
18074 }

```

Similarly for `almtreehypergroup`.


```

18075 \ifdef{\@glsstyle@almtreehypergroup}
18076 {%
18077   \renewglossarystyle{almtreehypergroup}{%
18078     \setglossarystyle{almtree}%
18079     \renewcommand*\glossaryheader{%
18080       \par
18081       \def\@gls@prevlevel{-1}%
18082       \hangindent0pt\relax
18083       \parindent0pt\relax
18084       \glstreenavigationfmt{\glsnavigation}%

```

Can't use \@afterheading here as it messes with the first item of the group.

```

18085     \glstreegroupheaderskip
18086   }%
18087   \renewcommand*\glsgroupheading}[1]{%
18088     \glstrgetgrouptitle{##1}{\glstr@grptitle}%
18089     \glstreePreHeader{##1}{\glstr@grptitle}%
18090     \par
18091     \def\@gls@prevlevel{-1}%
18092     \hangindent0pt\relax
18093     \parindent0pt\relax
18094     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glstr@grptitle}}}%

```

Can't use \@afterheading here as it messes with the first item of the group.

```

18095     \glstreegroupheaderskip
18096   }%
18097 }
18098 }%
18099 {%
18100 }

```

2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

18101 \ifdef{\@glsstyle@mcolindexgroup}
18102 {%
18103   \renewglossarystyle{mcolindexgroup}{%
18104     \setglossarystyle{mcolindex}%
18105     \renewcommand*\glsgroupheading}[1]{%
18106       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
18107       \glstreePreHeader{##1}{\glstr@grptitle}%
18108       \item\glstreegroupheaderfmt{\glstr@grptitle}%
18109       \glstreegroupheaderskip\@afterheading
18110     }%
18111   }
18112 }%
18113 {%
18114 }

```

Similarly for mcolindexhypergroup.

```

18115 \ifdef{\@glsstyle@mcolindexhypergroup}
18116 {%
18117   \renewglossarystyle{mcolindexhypergroup}{%
18118     \setglossarystyle{mcolindex}%
18119     \renewcommand*{\glossaryheader}{%
18120       \item\glstreenavigationfmt{\glsnavigation}%

18121       \glstreegroupheaderskip\@afterheading
18122     }%
18123     \renewcommand*{\glsgroupheading}[1]{%
18124       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
18125       \glstreePreHeader{##1}{\glstr@grptitle}%
18126       \item\glstreegroupheaderfmt
18127         {\glsnavhypertarget{##1}{\glstr@grptitle}}%
18128       \glstreegroupheaderskip\@afterheading
18129     }%
18130   }
18131 }%
18132 {%
18133 }

```

Similarly for mcolindexspannav.

```

18134 \ifdef{\@glsstyle@mcolindexspannav}
18135 {%
18136   \renewglossarystyle{mcolindexspannav}{%
18137     \setglossarystyle{index}%
18138     \renewenvironment{theglossary}%
18139     {%
18140       \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]{%
18141         \setlength{\parindent}{0pt}%
18142         \setlength{\parskip}{0pt plus 0.3pt}%
18143         \let\item\glstreeitem}%
18144       {\end{multicols}}%
18145     \renewcommand*{\glsgroupheading}[1]{%
18146       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
18147       \glstreePreHeader{##1}{\glstr@grptitle}%
18148       \item\glstreegroupheaderfmt
18149         {\glsnavhypertarget{##1}{\glstr@grptitle}}%
18150       \glstreegroupheaderskip\@afterheading
18151     }%
18152   }
18153 }%
18154 {%
18155 }

```

Similarly for mcoltreegroup.

```

18156 \ifdef{\@glsstyle@mcoltreegroup}
18157 {%
18158   \renewglossarystyle{mcoltreegroup}{%

```

```

18159 \setglossarystyle{mcoltree}%
18160 \renewcommand{\glsgroupheading}[1]{%
18161 \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
18162 \glstreePreHeader{##1}{\glxtr@grptitle}%
18163 \par\noindent\glstreegroupheaderfmt{\glxtr@grptitle}%
18164 \glstreegroupheaderskip\@afterheading
18165 }%
18166 }
18167 }%
18168 {%
18169 }

```

Similarly for mcoltreehypergroup.

```

18170 \ifdef{\@glsstyle@mcoltreehypergroup}
18171 {%
18172 \renewglossarystyle{mcoltreehypergroup}{%
18173 \setglossarystyle{mcoltree}%
18174 \renewcommand*\glsgroupheading{%
18175 \par\noindent\glstreenavigationfmt{\glsnavigation}%
18176 \glstreegroupheaderskip
18177 }%
18178 \renewcommand*\glsgroupheading}[1]{%
18179 \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
18180 \glstreePreHeader{##1}{\glxtr@grptitle}%
18181 \par\noindent
18182 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}%
18183 \glstreegroupheaderskip\@afterheading
18184 }%
18185 }
18186 }%
18187 {%
18188 }

```

Similarly for mcoltreespannav.

```

18189 \ifdef{\@glsstyle@mcoltreespannav}
18190 {%
18191 \renewglossarystyle{mcoltreespannav}{%
18192 \setglossarystyle{tree}%
18193 \renewenvironment{theglossary}%
18194 {%
18195 \begin{multicols}{\glsmcols}%
18196 [\noindent\glstreenavigationfmt{\glsnavigation}]]%
18197 \setlength{\parindent}{0pt}%
18198 \setlength{\parskip}{0pt plus 0.3pt}%
18199 }%
18200 {\end{multicols}}}%
18201 \renewcommand*\glsgroupheading}[1]{%
18202 \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
18203 \glstreePreHeader{##1}{\glxtr@grptitle}%
18204 \par\noindent

```

```

18205     \glstreegroupheaderfmt{\glsnavigationtarget{##1}{\glstr@grptitle}}}%
18206     \glstreegroupheaderskip\@afterheading
18207 }%
18208 }
18209 }%
18210 {%
18211 }

```

Similarly for mcoltreenonamegroup.

```

18212 \ifdef{\@glsstyle@mcoltreenonamegroup}
18213 {%
18214   \renewglossarystyle{mcoltreenonamegroup}{%
18215     \setglossarystyle{mcoltreenoname}%
18216     \renewcommand{\glsgroupheading}[1]{%
18217       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
18218       \glstreePreHeader{##1}{\glstr@grptitle}%
18219       \par\noindent\glstreegroupheaderfmt{\glstr@grptitle}%
18220       \glstreegroupheaderskip\@afterheading
18221     }%
18222   }
18223 }%
18224 {%
18225 }

```

Similarly for mcoltreenonamehypergroup.

```

18226 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
18227 {%
18228   \renewglossarystyle{mcoltreenonamehypergroup}{%
18229     \setglossarystyle{mcoltreenoname}%
18230     \renewcommand*\{glossaryheader}{%
18231       \par\noindent\glstreenavigationfmt{\glsnavigation}%
18232       \glstreegroupheaderskip
18233     }%
18234     \renewcommand*\{glsgroupheading}[1]{%
18235       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
18236       \glstreePreHeader{##1}{\glstr@grptitle}%
18237       \par\noindent
18238       \glstreegroupheaderfmt{\glsnavigationtarget{##1}{\glstr@grptitle}}}%
18239       \glstreegroupheaderskip\@afterheading}%
18240   }
18241 }%
18242 {%
18243 }

```

Similarly for mcoltreenonamespannav.

```

18244 \ifdef{\@glsstyle@mcoltreenonamespannav}
18245 {%
18246   \renewglossarystyle{mcoltreenonamespannav}{%
18247     \setglossarystyle{treenoname}%
18248     \renewenvironment{theglossary}%
18249     {%

```

```

18250     \begin{multicols}{\glsmcols}%
18251     [\noindent\glstreenavigationfmt{\glsnavigation}]%
18252     \setlength{\parindent}{0pt}%
18253     \setlength{\parskip}{0pt plus 0.3pt}%
18254 }%
18255 {\end{multicols}}%
18256 \renewcommand*{\glsgroupheading}[1]{%
18257     \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
18258     \glstreePreHeader{##1}{\glxtr@grptitle}%
18259     \par\noindent
18260     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}%
18261     \glstreegroupheaderskip\@afterheading}%
18262 }
18263 }%
18264 {%
18265 }

```

`mcولاتree` needs adjusting so that it uses `\glxtralttreeInit`. This doesn't use `\mbox{\par` which would unbalance the top of the columns.

```

18266 \ifdef{\@glsstyle@mcولاتree}
18267 {%
18268     \renewglossarystyle{mcولاتree}{%
18269         \setglossarystyle{alttree}%
18270         \renewenvironment{theglossary}%
18271         {%
18272             \glxtralttreeInit
18273             \def\@gls@prevlevel{-1}%
18274             \begin{multicols}{\glsmcols}%
18275             }%
18276             {\par\end{multicols}}%
18277         }
18278     }%
18279 {%
18280 }

```

Redefine `mcولاتreegroup` to discourage page breaks after the group headings.

```

18281 \ifdef{\@glsstyle@mcولاتreegroup}
18282 {%
18283     \renewglossarystyle{mcولاتreegroup}{%
18284         \setglossarystyle{mcولاتree}%
18285         \renewcommand{\glsgroupheading}[1]{%
18286             \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
18287             \glstreePreHeader{##1}{\glxtr@grptitle}%
18288             \par
18289             \def\@gls@prevlevel{-1}%
18290             \hangindentOpt\relax
18291             \parindentOpt\relax
18292             \glstreegroupheaderfmt{\glxtr@grptitle}%
18293             \glstreegroupheaderskip
18294         }%

```

```

18295 }
18296 }%
18297 {%
18298 }

```

Similarly for mcolalttreehypergroup.

```

18299 \ifdef{\@glsstyle@mcolalttreehypergroup}
18300 {%
18301   \renewglossarystyle{mcolalttreehypergroup}{%
18302     \setglossarystyle{mcolalttree}%
18303     \renewcommand*{\glossaryheader}{%
18304       \par
18305       \def\@gls@prevlevel{-1}%
18306       \hangindent0pt\relax
18307       \parindent0pt\relax
18308       \glstreenavigationfmt{\glsnavigation}%
18309       \glstreegroupheaderskip
18310     }%
18311     \renewcommand*{\glsgroupheading}[1]{%
18312       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
18313       \glstreePreHeader{##1}{\glxtr@grptitle}%
18314       \par
18315       \def\@gls@prevlevel{-1}%
18316       \hangindent0pt\relax
18317       \parindent0pt\relax
18318       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}%
18319       \glstreegroupheaderskip
18320     }%
18321   }
18322 }%
18323 {%
18324 }

```

Similarly for mcolalttreespannav.

```

18325 \ifdef{\@glsstyle@mcolalttreespannav}
18326 {%
18327   \renewglossarystyle{mcolalttreespannav}{%
18328     \setglossarystyle{alttree}%
18329     \renewenvironment{theglossary}%
18330     {%
18331       \glxtralttreeInit
18332       \def\@gls@prevlevel{-1}%
18333       \begin{multicols}{\glsmcols}%
18334         [\noindent\glstreenavigationfmt{\glsnavigation}]%
18335     }%
18336     {\par\end{multicols}}%
18337     \renewcommand*{\glsgroupheading}[1]{%
18338       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
18339       \glstreePreHeader{##1}{\glxtr@grptitle}%
18340     \par

```

```

18341      \def\@gls@prevlevel{-1}%
18342      \hangindent0pt\relax
18343      \parindent0pt\relax
18344      \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\@glsxtr@grptitle}}}%
18345      \glstreegroupheaderskip
18346    }%
18347  }
18348 }%
18349 {%
18350 }

      Reset the default style
18351 \ifx\@glossary@default@style\relax
18352 \else
18353   \setglossarystyle{\@glsxtr@current@style}
18354 \fi

```

3 bookindex style (glossary-bookindex.sty)

3.1 Package Initialisation and Options

```
18355 \NeedsTeXFormat{LaTeX2e}
18356 \ProvidesPackage{glossary-bookindex}[2021/11/22 v1.48 (NLCT)]
```

Load required packages.

```
18357 \RequirePackage{multicol}
18358 \RequirePackage{glossary-tree}
```

trbookindexcols Number of columns.

```
18359 \newcommand{\glstrbookindexcols}{2}
```

trbookindexname Format used for top-level entries. (Argument is the label.)

```
18360 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

ookindexsubname Format used for sub entries.

```
18361 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

sxtrprelocation Provide in case glossaries-stylemods isn't loaded.

```
18362 \providecommand*\glsxtrprelocation{\space}
```

ndexprelocation Separator used before location list for top-level entries. Version 1.22 has removed the \ifglsnopostdot check since this style doesn't display the description.

```
18363 \newcommand*{\glsxtrbookindexprelocation}[1]{%
18364   \glxtrifhasfield{location}{#1}%
18365   {,\glsxtrprelocation}%
18366   {\glsxtrprelocation}%
18367 }
```

xsubprelocation Separator used before location list for sub-entries.

```
18368 \newcommand*{\glsxtrbookindexsubprelocation}[1]{%
18369   \glsxtrbookindexprelocation{#1}%
18370 }
```

okindexlocation

```
\glsxtrbookindexlocation{\label}{\location}
```

Displays the location.

```
18371 \newcommand*{\glsxtrbookindexlocation}[2]{#2}
```


ndexsublocation

```
\glxtrbookindexlocation{<label>}{<location>}
```

Displays the location for sub-entries.

```
18372 \newcommand*{\glxtrbookindexsublocation}{\glxtrbookindexlocation}
```

xparentchildsep Separator used between top-level parent and child entry.

```
18373 \newcommand{\glxtrbookindexparentchildsep}{\nopagebreak}
```

rentsubchildsep Separator used between sub-level parent and child entry.

```
18374 \newcommand{\glxtrbookindexparentsubchildsep}{\glxtrbookindexparentchildsep}
```

ookindexbetween Between two top-level entries identified by the labels in the arguments.

```
18375 \newcommand{\glxtrbookindexbetween}[2]{}
```

indexsubbetween Between two level 1 entries identified by the labels in the arguments.

```
18376 \newcommand{\glxtrbookindexsubbetween}[2]{}
```

exsubsubbetween Between two level 2 entries identified by the labels in the arguments.

```
18377 \newcommand{\glxtrbookindexsubsubbetween}[2]{}
```

indexatendgroup At the end of a letter group. The argument is the index of the last top-level entry.

```
18378 \newcommand{\glxtrbookindexatendgroup}[1]{}
```

exsubatendgroup At the end of a letter group. The argument is the index of the last level 1 entry.

```
18379 \newcommand{\glxtrbookindexsubatendgroup}[1]{}
```

subsubatendgroup At the end of a letter group. The argument is the index of the last level 2 entry.

```
18380 \newcommand{\glxtrbookindexsubsubatendgroup}[1]{}
```

kindexgroupskip Group separator.

```
18381 \newcommand{\glxtrbookindexgroupskip}{\ifglxnogroupskip\else\indexspace\fi}
```

Format group title.

dexformatheader Group separator.

```
18382 \newcommand*{\glxtrbookindexformatheader}[1]{%
18383   \par{\centering\glstreegroupheaderfmt{#1}\par}%
18384 }
```

okindexbookmark Book mark group heading if supported.

```
18385 \ifdef\pdfbookmark
18386 {%
18387   \newcommand*{\glxtrbookindexbookmark}[2]{%
18388     \ifdefstring{@@glossarysec}{chapter}%
18389     {\pdfbookmark[1]{#1}{#2}}%
18390   }
18391 }
```

```

18390     {\pdfbookmark[2]{#1}{#2}}%
18391   }
18392 }
18393 {%
18394   \newcommand*{\glxtrbookindexbookmark}[2]{%
18395 }

```

bookmarkprefix Make the bookmark label prefix used for letter groups depend on the glossary label (instead of original hardcoded “index.”).

```

18396 \newcommand*{\glxtrbookindexbookmarkprefix}{\currentglossary.}

```

indexcolspread

```

18397 \newcommand*{\glxtrbookindexcolspread}{}

```

indexmulticolseenv

```

18398 \newcommand*{\glxtrbookindexmulticolseenv}{multicols}

```

Define the style.

```

18399 \newglossarystyle{bookindex}{%
18400   \setglossarystyle{index}%
18401   \renewenvironment{theglossary}%
18402   {%
18403     \ifnum\glxtrbookindexcols>1\relax
18404       \ifdefempty\glxtrbookindexcolspread
18405       {%
18406         \edef\glxtr@beginbookindex{%
18407           \noexpand\begin{\glxtrbookindexmulticolseenv}
18408             {\glxtrbookindexcols}%
18409         }%
18410       }%
18411     {%
18412       \edef\glxtr@beginbookindex{%
18413         \noexpand\begin{\glxtrbookindexmulticolseenv}%
18414           {\glxtrbookindexcols}{\glxtrbookindexcolspread}%
18415       }%
18416     }%
18417   \else
18418     \def\glxtr@beginbookindex{%
18419   \fi
18420   \glxtr@beginbookindex
18421   \setlength{\parindent}{0pt}%
18422   \setlength{\parskip}{0pt plus 0.3pt}%
18423   \let\@glxtr@bookindex@sep\glxtrbookindexparentchildsep
18424   \let\@glxtr@bookindex@subsep\glxtrbookindexparentsubchildsep
18425   \let\@glxtr@bookindex@between\@gobble
18426   \let\@glxtr@bookindex@subbetween\@gobble
18427   \let\@glxtr@bookindex@subsubbetween\@gobble
18428   \let\@glxtr@bookindex@atendgroup\relax
18429   \let\@glxtr@bookindex@subatendgroup\relax

```

```

18430 \let\@glstr@bookindex@subsubatendgroup\relax
18431 \let\@glstr@bookindexgroupskip\relax
18432 }%
18433 {%

```

Do end group hooks.

```

18434 \@glstr@bookindex@subsubatendgroup
18435 \@glstr@bookindex@subatendgroup
18436 \@glstr@bookindex@atendgroup

```

End multicol environment.

```

18437 \ifnum\glstrbookindexcols>1\relax
18438 \edef\glstr@endbookindex{%
18439 \noexpand\end{\glstrbookindexmulticolenv}%
18440 }%
18441 \else
18442 \def\glstr@endbookindex{%
18443 \fi
18444 \glstr@endbookindex
18445 }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

18446 \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

18447 \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

18448 \@glstr@bookindex@between{##1}%

```

Update separators.

```

18449 \let\@glstr@bookindex@sep\glstrbookindexparentchildsep
18450 \let\@glstr@bookindex@subsep\glstrbookindexparentsubchildsep
18451 \let\@glstr@bookindex@subbetween\@gobble
18452 \let\@glstr@bookindex@subsubbetween\@gobble
18453 \edef\@glstr@bookindex@between{%
18454 \noexpand\glstrbookindexbetween{##1}%
18455 }%
18456 \edef\@glstr@bookindex@atendgroup{%
18457 \noexpand\glstrbookindexatendgroup{##1}%
18458 }%
18459 \let\@glstr@bookindex@subatendgroup\relax
18460 \let\@glstr@bookindex@subsubatendgroup\relax

```

Format entry.

```

18461 \glstreeitem
18462 \glstryitem{##1}%
18463 \glstarget{##1}{\glstrbookindexname{##1}}%
18464 \glstrbookindexprelocation{##1}%
18465 \glstrbookindexlocation{##1}{##2}%
18466 }%
18467 \renewcommand{\subglossentry}[3]{%
18468 \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```
18469      \glstreeitem
18470      \or
```

Level 1.

```
18471      \@glstr@bookindex@sep
18472      \@glstr@bookindex@subbetween{##2}%
18473      \let\@glstr@bookindex@sep\relax
```

Update separators.

```
18474      \let\@glstr@bookindex@subsubbetween\@gobble
18475      \let\@glstr@bookindex@subsep\glstrbookindexparentsubchildsep
18476      \edef\@glstr@bookindex@subbetween{%
18477          \noexpand\glstrbookindexsubbetween{##2}%
18478      }%
18479      \edef\@glstr@bookindex@atsubendgroup{%
18480          \noexpand\glstrbookindexatsubendgroup{##1}%
18481      }%
```

Start sub-item.

```
18482      \glstreesubitem
18483      \glssubentryitem{##2}%
18484      \else
```

All other levels.

```
18485      \@glstr@bookindex@subsep
18486      \@glstr@bookindex@subsubbetween{##2}%
```

Update separators.

```
18487      \let\@glstr@bookindex@subsep\relax
18488      \edef\@glstr@bookindex@subsubbetween{%
18489          \noexpand\glstrbookindexsubsubbetween{##2}%
18490      }%
18491      \edef\@glstr@bookindex@atsubsubendgroup{%
18492          \noexpand\glstrbookindexatsubsubendgroup{##1}%
18493      }%
```

Start sub-sub-item.

```
18494      \glstreesubsubitem
18495      \fi
```

Format entry.

```
18496      \glstarget{##2}{\glstrbookindexsubname{##2}}%
18497      \glstrbookindexsubprelocation{##2}%
18498      \glstrbookindexsublocation{##2}{##3}%
18499      }%
```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
18500      \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
18501      \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
18502 \glxtr@bookindex@subsubatendgroup
18503 \glxtr@bookindex@subatendgroup
18504 \glxtr@bookindex@atendgroup
18505 \glxtr@bookindexgroupskip
```

Update separators.

```
18506 \let\glxtr@bookindexgroupskip\glxtrbookindexgroupskip
18507 \let\glxtr@bookindex@between\@gobble
18508 \let\glxtr@bookindex@atendgroup\relax
18509 \let\glxtr@bookindex@subatendgroup\relax
18510 \let\glxtr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
18511 \glxtrgetgrouptitle{##1}{\glxtrcurrentgrptitle}%
```

Do the PDF bookmark if supported.

```
18512 \glxtrbookindexbookmark{\glxtrcurrentgrptitle}{\glxtrbookindexbookmarkprefix##1}%
```

Format the group title.

```
18513 \glxtrbookindexformatheader{\glxtrcurrentgrptitle}%
18514 \nopagebreak\indexspace\nopagebreak\@afterheading
18515 }%
18516 }
```

Some supplementary commands that may be useful. These store the entry label for the current page. Since the page number is needed in the control sequence, this uses `\glxtrbookindexthepage` instead of `\thepage` in case the page numbering has been set to something that contains formatting commands.

`bookindexthepage` The `\@printglossary` sets `\currentglossary` to the current glossary label. This is used as a prefix in case the page number is reset.

```
18517 \newcommand{\glxtrbookindexthepage}{%
18518 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
18519 }
```

`bookindexmarkentry` Writes entry information to the `.aux` file. The argument is the entry label.

```
18520 \newcommand*{\glxtrbookindexmarkentry}[1]{%
18521 \protected@write\@auxout
18522 {\let\glxtrbookindexthepage\relax}%
18523 {\string\glxtr@setbookindexmark{\glxtrbookindexthepage}{#1}}%
18524 }
```

`setbookindexmark`

```
18525 \newcommand*{\glxtr@setbookindexmark}[2]{%
18526 \ifcsundef\glxtr@idxfirstmark@#1}%
18527 {\csgdef\glxtr@idxfirstmark@#1}{#2}}%
18528 {}%
18529 \csgdef\glxtr@idxlastmark@#1}{#2}%
18530 }
```

dexfirstmarkfmt

```
18531 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
18532   \glstryname{#1}%
18533 }
```

kindexfirstmark

```
18534 \newcommand*{\glxtrbookindexfirstmark}{%
18535   \letcs{\glxtr@label}{\glxtr@idxfirstmark@\glxtrbookindexthepage}%
18536   \ifdef\glxtr@label
18537     {\glxtrbookindexfirstmarkfmt{\glxtr@label}}%
18538     {}%
18539 }
```

ndexlastmarkfmt

```
18540 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
18541   \glstryname{#1}%
18542 }
```

okindexlastmark

```
18543 \newcommand*{\glxtrbookindexlastmark}{%
18544   \letcs{\glxtr@label}{\glxtr@idxlastmark@\glxtrbookindexthepage}%
18545   \ifdef\glxtr@label
18546     {\glxtrbookindexlastmarkfmt{\glxtr@label}}%
18547     {}%
18548 }
```

4 longextra styles (glossary-longextra.sty)

4.1 Package Initialisation and Options

Provides additional long styles.

```
18549 \NeedsTeXFormat{LaTeX2e}  
18550 \ProvidesPackage{glossary-longextra}[2021/11/22 v1.48 (NLCT)]
```

Load required packages.

```
18551 \RequirePackage{glossary-longbooktabs}
```

ongextraNameFmt

```
\glslongextraNameFmt{<label>}
```

Governs the way the name is displayed.

```
18552 \newcommand{\glslongextraNameFmt}[1]{%  
18553   \glsentryitem{#1}\glstarget{#1}{\glossentryname{#1}}%  
18554 }
```

ongextraDescFmt

```
\glslongextraDescFmt{<label>}
```

Governs the way the description is displayed.

```
18555 \newcommand{\glslongextraDescFmt}[1]{%  
18556   \glossentrydesc{#1}\glspostdescription  
18557 }
```

gextraSymbolFmt

```
\glslongextraSymbolFmt{<label>}
```

Governs the way the symbol is displayed.

```
18558 \newcommand{\glslongextraSymbolFmt}[1]{\glossentrysymbol{#1}}
```

extraLocationFmt

```
\glslongextraLocationFmt{<label>}{<location list>}
```

Governs the way the location is displayed.

```
18559 \newcommand{\glslongextraLocationFmt}[2]{#2}
```

extraSubNameFmt

```
\glslongextraSubNameFmt{<level>}{<label>}
```

Governs the way the child name is displayed. Just does the sub-entry counter, if enabled, and the target.

```
18560 \newcommand{\glslongextraSubNameFmt}[2]{%  
18561 \glssubentryitem{#2}\glstarget{#2}{\strut}%  
18562 }
```

extraSubDescFmt

```
\glslongextraSubDescFmt{<level>}{<label>}
```

Governs the way the child description is displayed.

```
18563 \newcommand{\glslongextraSubDescFmt}[2]{%  
18564 \glslongextraDescFmt{#2}%  
18565 }
```

extraSubSymbolFmt

```
\glslongextraSubSymbolFmt{<level>}{<label>}
```

Governs the way the child symbol is displayed.

```
18566 \newcommand{\glslongextraSubSymbolFmt}[2]{%  
18567 \glslongextraSymbolFmt{#2}%  
18568 }
```

extraSubLocationFmt

```
\glslongextraSubLocationFmt{<level>}{<label>}{<location list>}
```

Governs the way the child location list is displayed.

```
18569 \newcommand{\glslongextraSubLocationFmt}[3]{#3}
```


`gextraNameAlign` Alignment for the name column.
18570 `\newcommand{\glslongextraNameAlign}{l}`

`gextraDescAlign` Alignment for the description column.
18571 `\newcommand{\glslongextraDescAlign}{>{\raggedright}p{\glstdescwidth}}`

`extraSymbolAlign` Alignment for the symbol column.
18572 `\newcommand{\glslongextraSymbolAlign}{c}`

`raLocationAlign` Alignment for the location column.
18573 `\newcommand{\glslongextraLocationAlign}{>{\raggedright}p{\glspagelistwidth}}`

`extraGroupHeading` Used to format the letter group headings. The first argument is the number of columns in the table. The second is the group *label* (not the title).
18574 `\newcommand{\glslongextraGroupHeading}[2]{}`

`extraHeaderFormat` Format for the column headers.
18575 `\newcommand{\glslongextraHeaderFmt}[1]{\textbf{#1}}`

`aNameDescHeader`
18576 `\newcommand{\glslongextraNameDescHeader}{%`
18577 `\glslongextraNameDescTabularHeader\endhead`
18578 `\glslongextraNameDescTabularFooter\endfoot`
18579 `}`

`scTabularHeader`
18580 `\newcommand{\glslongextraNameDescTabularHeader}{%`
18581 `\toprule`
18582 `\glslongextraHeaderFmt\entryname &`
18583 `\glslongextraHeaderFmt\descriptionname\tabularnewline`
18584 `\midrule`
18585 `}`

`scTabularFooter`
18586 `\newcommand{\glslongextraNameDescTabularFooter}{%`
18587 `\bottomrule`
18588 `}`

Unlike the `alttree` style, there aren't different widths for the hierarchical levels.

`gextraSetWidest` Provide in case the tree styles haven't been loaded.
18589 `\newcommand*{\glslongextraSetWidest}[1]{%`
18590 `\def\@glslongextrawidestname{#1}%`
18591 `}`

`extrawidestname` Pick up the widest name from the `alttree` style if it has been set. (Will expand to nothing otherwise.)
18592 `\newcommand*{\@glslongextrawidestname}{\csuse{@glswidestname}}`

traUpdateWidest

```

18593 \newcommand*{\glslongextraUpdateWidest}[1]{%
18594   \ifundef\@glslongextrawidestname
18595   {\def\@glslongextrawidestname{#1}}%
18596   {%
18597     \settowidth{\dimen@}{\@glslongextrawidestname}%
18598     \settowidth{\dimen@ii}{#1}%
18599     \ifdim\dimen@ii>\dimen@
18600     \def\@glslongextrawidestname{#1}%
18601     \fi
18602   }%
18603 }
```

dateWidestChild

`\glslongextraUpdateWidestChild{<level>}{<text>}`

Used by `\glsxtrSetWidest` in `glossaries-extra-bib2gls`. Does nothing by default, since the default action in these styles is to omit the child name. If the child name should be displayed, then this needs to be redefined to use `\glslongextraUpdateWidest`.

```
18604 \newcommand*{\glslongextraUpdateWidestChild}[2]{}
```

traSetDescWidth Computes the value of `\glsdescwidth` for the styles that only have name and description columns.

```

18605 \newcommand{\glslongextraSetDescWidth}{%
18606   \settowidth{\gls@tmplen}{\glslongextraHeaderFmt\entryname}%

   Has the widest name been set.

18607   \settowidth{\dimen@}{\glsnamefont{\@glslongextrawidestname}}%
18608   \ifdim\dimen@>\gls@tmplen
18609     \gls@tmplen=\dimen@
18610   \fi
```

Description width is `\linewidth` less `4\tabcolsep` less the width of the name column.

```

18611   \setlength{\glsdescwidth}{\dimexpr\linewidth-4\tabcolsep-\gls@tmplen}%
18612 }
```

SymSetDescWidth Computes the value of `\glsdescwidth` for the styles that only have name, symbol and description columns.

```
18613 \newcommand{\glslongextraSymSetDescWidth}{%
```

Work out the size for just the name and description style.

```
18614   \glslongextraSetDescWidth
```

Now work out the symbol column width. This is assuming that the column title will be the widest text in the column.

```
18615   \settowidth{\gls@tmplen}{\glslongextraHeaderFmt\symbolname}%
```

Subtract 2\tabcolsep and the symbol header width.

```

18616 \setlength{\glsdescwidth}{\dimexpr\glsdescwidth-2\tabcolsep-\gls@tmplen}%
18617 }

```

LocSetDescWidth Computes the value of \glsdescwidth for the styles that only have name, location and description columns.

```

18618 \newcommand{\glslongextraLocSetDescWidth}{%
  Work out the size for just the name and description style.
18619 \glslongextraSetDescWidth
  Subtract 2\tabcolsep and the location list column width.
18620 \setlength{\glsdescwidth}{\dimexpr\glsdescwidth-2\tabcolsep-\glspagelistwidth}%
18621 }

```

LocSetDescWidth Computes the value of \glsdescwidth for the styles that have name, symbol, location and description columns.

```

18622 \newcommand{\glslongextraSymLocSetDescWidth}{%
  Work out the size for just the name, symbol and description style.
18623 \glslongextraSymSetDescWidth
  Subtract 2\tabcolsep and the location list column width.
18624 \setlength{\glsdescwidth}{\dimexpr\glsdescwidth-2\tabcolsep-\glspagelistwidth}%
18625 }

```

ExtraUseTabular If true use tabular instead of longtable. Obviously only intended for short glossaries that can fit into a single page.

```

18626 \newif\ifGlsLongExtraUseTabular
18627 \GlsLongExtraUseTabularfalse

```

raTabularVAlign Only used with the tabular setting.

```

18628 \newcommand*{\glslongextraTabularVAlign}{c}

```

long-name-desc Two column style with multi-lined descriptions and header. This is similar to the longragged-booktabs style.

```

18629 \newglossarystyle{long-name-desc}%
18630 {%
18631 \ifGlsLongExtraUseTabular
18632 \renewenvironment{theglossary}%
18633 {%
18634 \glslongextraSetDescWidth
18635 \edef\@glslongextra@begintab{%
18636 \noexpand\begin{tabular}{\glslongextraTabularVAlign}{%
18637 \expandonce\glslongextraNameAlign
18638 \expandonce\glslongextraDescAlign}}%
18639 \@glslongextra@begintab
18640 }%
18641 {%

```

```

18642     \glslongextraNameDescTabularFooter
18643     \end{tabular}%
18644 }%
18645 \renewcommand*{\glossaryheader}{\glslongextraNameDescTabularHeader}%
18646 \else
18647 \renewenvironment{theglossary}%
18648 {%
18649     \glspatchLToutput
18650     \glslongextraSetDescWidth
18651     \edef\@glslongextra@begintab{%
18652         \noexpand\begin{longtable}{%
18653             \expandonce\glslongextraNameAlign
18654             \expandonce\glslongextraDescAlign}}%
18655     \@glslongextra@begintab
18656 }%
18657 {\end{longtable}}%
18658 \renewcommand*{\glossaryheader}{\glslongextraNameDescHeader}%
18659 \fi
18660 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{2}{##1}}%
18661 \renewcommand{\glossentry}[2]{%
18662     \glslongextraNameFmt{##1} &
18663     \glslongextraDescFmt{##1}\tabularnewline
18664 }%
18665 \renewcommand{\subglossentry}[3]{%
18666     \glslongextraSubNameFmt{##1}{##2}
18667     &
18668     \glslongextraSubDescFmt{##1}{##2}%
18669     \tabularnewline
18670 }%
18671 \ifglsnogroupskip
18672     \renewcommand*{\glsgroupskip}{}%
18673 \else
18674     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
18675 \fi
18676 }

```

cLocationHeader

```

18677 \newcommand{\glslongextraNameDescLocationHeader}{%
18678     \glslongextraNameDescLocationTabularHeader\endhead
18679     \glslongextraNameDescLocationTabularFooter\endfoot
18680 }

```

onTabularHeader

```

18681 \newcommand{\glslongextraNameDescLocationTabularHeader}{%
18682     \toprule
18683     \glslongextraHeaderFmt\entryname &
18684     \glslongextraHeaderFmt\descriptionname &
18685     \glslongextraHeaderFmt\pagelistname\tabularnewline
18686     \midrule

```

18687 }

onTabularFooter

```
18688 \newcommand{\glslongextraNameDescLocationTabularFooter}{%
18689   \bottomrule
18690 }
```

g-name-desc-loc Three columns: name, description and location list.

```
18691 \newglossarystyle{long-name-desc-loc}%
18692 {%
18693   \ifGlsLongExtraUseTabular
18694     \renewenvironment{theglossary}%
18695       {%
18696         \glslongextraLocSetDescWidth
18697         \edef\@glslongextra@begintab{%
18698           \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
18699             \expandonce\glslongextraNameAlign
18700             \expandonce\glslongextraDescAlign
18701             \expandonce\glslongextraLocationAlign
18702           }}%
18703         \@glslongextra@begintab
18704       }%
18705     {%
18706       \glslongextraNameDescLocationTabularFooter
18707       \end{tabular}%
18708     }%
18709   \renewcommand*\glossaryheader{\glslongextraNameDescLocationTabularHeader}%
18710   \else
18711     \renewenvironment{theglossary}%
18712       {%
18713         \glspatchLToutput
18714         \glslongextraLocSetDescWidth
18715         \edef\@glslongextra@begintab{%
18716           \noexpand\begin{longtable}%
18717             \expandonce\glslongextraNameAlign
18718             \expandonce\glslongextraDescAlign
18719             \expandonce\glslongextraLocationAlign
18720           }}%
18721         \@glslongextra@begintab
18722       }%
18723     {\end{longtable}}%
18724   \renewcommand*\glossaryheader{\glslongextraNameDescLocationHeader}%
18725   \fi
18726   \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{3}{##1}}%
18727   \renewcommand{\glossentry}[2]{%
18728     \glslongextraNameFmt{##1} &
18729     \glslongextraDescFmt{##1} &
18730     \glslongextraLocationFmt{##1}{##2}\tabularnewline
18731   }%
```

```

18732 \renewcommand{\subglossentry}[3]{%
18733     \glslongextraSubNameFmt{##1}{##2}&
18734     \glslongextraSubDescFmt{##1}{##2}&
18735     \glslongextraSubLocationFmt{##1}{##2}{##3}%
18736     \tabularnewline
18737 }%
18738 \ifglsgnogroupskip
18739 \renewcommand*{\glsgroupskip}{}%
18740 \else
18741 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
18742 \fi
18743 }

```

aDescNameHeader

```

18744 \newcommand{\glslongextraDescNameHeader}{%
18745 \glslongextraDescNameTabularHeader\endhead
18746 \glslongextraDescNameTabularFooter\endfoot
18747 }

```

meTabularHeader

```

18748 \newcommand{\glslongextraDescNameTabularHeader}{%
18749 \toprule
18750 \glslongextraHeaderFmt\descriptionname&
18751 \glslongextraHeaderFmt\entryname \tabularnewline
18752 \midrule
18753 }

```

meTabularFooter

```

18754 \newcommand{\glslongextraDescNameTabularFooter}{%
18755 \bottomrule
18756 }

```

long-desc-name Like name-desc but swaps the columns.

```

18757 \newglossarystyle{long-desc-name}%
18758 {%
18759 \ifGlsLongExtraUseTabular
18760 \renewenvironment{theglossary}%
18761 {%
18762     \glslongextraSetDescWidth
18763     \edef\@glslongextra@begintab{%
18764         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
18765             \expandonce\glslongextraDescAlign
18766             \expandonce\glslongextraNameAlign}}%
18767     \@glslongextra@begintab
18768 }%
18769 {%
18770     \glslongextraDescNameTabularFooter
18771     \end{tabular}%
18772 }%

```

```

18773 \renewcommand*{\glossaryheader}{\glslongextraDescNameTabularHeader}%
18774 \else
18775 \renewenvironment{theglossary}%
18776 {%
18777 \glspatchLToutput
18778 \glslongextraSetDescWidth
18779 \edef\@glslongextra@begintab{%
18780 \noexpand\begin{longtable}{%
18781 \expandonce\glslongextraDescAlign
18782 \expandonce\glslongextraNameAlign}}%
18783 \@glslongextra@begintab
18784 }%
18785 {\end{longtable}}%
18786 \renewcommand*{\glossaryheader}{\glslongextraDescNameHeader}%
18787 \fi
18788 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{2}{##1}}%
18789 \renewcommand{\glossentry}[2]{%
18790 \glslongextraDescFmt{##1} &
18791 \glslongextraNameFmt{##1}\tabularnewline
18792 }%
18793 \renewcommand{\subglossentry}[3]{%
18794 \glslongextraSubDescFmt{##1}{##2} &
18795 \glslongextraSubNameFmt{##1}{##2}\tabularnewline
18796 }%
18797 \ifglsnogroupskip
18798 \renewcommand*{\glsgroupskip}{}%
18799 \else
18800 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
18801 \fi
18802 }

```

nDescNameHeader

```

18803 \newcommand{\glslongextraLocationDescNameHeader}{%
18804 \glslongextraLocationDescNameTabularHeader\endhead
18805 \glslongextraLocationDescNameTabularFooter\endfoot
18806 }

```

meTabularHeader

```

18807 \newcommand{\glslongextraLocationDescNameTabularHeader}{%
18808 \toprule
18809 \glslongextraHeaderFmt\pagelistname&
18810 \glslongextraHeaderFmt\descriptionname&
18811 \glslongextraHeaderFmt\entryname \tabularnewline
18812 \midrule
18813 }

```

meTabularFooter

```

18814 \newcommand{\glslongextraLocationDescNameTabularFooter}{%
18815 \bottomrule

```

18816 }

g-loc-desc-name Three columns: location, description and name.

```
18817 \newglossarystyle{long-loc-desc-name}%
18818 {%
18819   \ifGlsLongExtraUseTabular
18820     {%
18821       \glslongextraLocSetDescWidth
18822       \edef\@glslongextra@begintab{%
18823         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
18824           \expandonce\glslongextraLocationAlign
18825           \expandonce\glslongextraDescAlign
18826           \expandonce\glslongextraNameAlign}}%
18827       \@glslongextra@begintab
18828     }%
18829     {%
18830       \glslongextraLocationDescNameTabularFooter
18831       \end{tabular}%
18832     }%
18833     \renewcommand*\glossaryheader{\glslongextraLocationDescNameTabularHeader}%
18834   \else
18835     \renewenvironment{theglossary}%
18836     {%
18837       \glspatchLToutput
18838       \glslongextraLocSetDescWidth
18839       \edef\@glslongextra@begintab{%
18840         \noexpand\begin{longtable}{%
18841           \expandonce\glslongextraLocationAlign
18842           \expandonce\glslongextraDescAlign
18843           \expandonce\glslongextraNameAlign}}%
18844       \@glslongextra@begintab
18845     }%
18846     {\end{longtable}}%
18847     \renewcommand*\glossaryheader{\glslongextraLocationDescNameHeader}%
18848   \fi
18849   \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{3}{##1}}%
18850   \renewcommand{\glossentry}[2]{%
18851     \glslongextraLocationFmt{##1}{##2} &
18852     \glslongextraDescFmt{##1} &
18853     \glslongextraNameFmt{##1}\tabularnewline
18854   }%
18855   \renewcommand{\subglossentry}[3]{%
18856     \glslongextraSubLocationFmt{##1}{##2}{##3} &
18857     \glslongextraSubDescFmt{##1}{##2} &
18858     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
18859   }%
18860   \ifglsgroupskip
18861     \renewcommand*\glsgroupskip{}%
18862   \else
```



```

18863 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
18864 \fi
18865 }

```

meDescSymHeader

```

18866 \newcommand{\glslongextraNameDescSymHeader}{%
18867 \glslongextraNameDescSymTabularHeader\endhead
18868 \glslongextraNameDescSymTabularFooter\endfoot
18869 }

```

ymTabularHeader

```

18870 \newcommand{\glslongextraNameDescSymTabularHeader}{%
18871 \toprule
18872 \glslongextraHeaderFmt\entryname &
18873 \glslongextraHeaderFmt\descriptionname &
18874 \glslongextraHeaderFmt\symbolname\tabularnewline
18875 \midrule
18876 }

```

ymTabularFooter

```

18877 \newcommand{\glslongextraNameDescSymTabularFooter}{%
18878 \bottomrule
18879 }

```

g-name-desc-sym Three column style with symbol in the third column.

```

18880 \newglossarystyle{long-name-desc-sym}%
18881 {%
18882 \ifGlsLongExtraUseTabular
18883 \renewenvironment{theglossary}%
18884 {%
18885 \glslongextraSymSetDescWidth
18886 \edef\@glslongextra@begintab{%
18887 \noexpand\begin{tabular}{\glslongextraTabularVAlign}{%
18888 \expandonce\glslongextraNameAlign
18889 \expandonce\glslongextraDescAlign
18890 \expandonce\glslongextraSymbolAlign
18891 }}%
18892 \@glslongextra@begintab
18893 }%
18894 {%
18895 \glslongextraNameDescSymTabularFooter
18896 \end{tabular}%
18897 }%
18898 \renewcommand*{\glossaryheader}{\glslongextraNameDescSymTabularHeader}%
18899 \else
18900 \renewenvironment{theglossary}%
18901 {%
18902 \glspatchLToutput
18903 \glslongextraSymSetDescWidth

```

```

18904 \edef\@glslongextra@begintab{%
18905 \noexpand\begin{longtable}{%
18906 \expandonce\glslongextraNameAlign
18907 \expandonce\glslongextraDescAlign
18908 \expandonce\glslongextraSymbolAlign
18909 }}%
18910 \@glslongextra@begintab
18911 }%
18912 {\end{longtable}}}%
18913 \renewcommand*\glossaryheader{\glslongextraNameDescSymHeader}%
18914 \fi
18915 \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{3}{##1}}%
18916 \renewcommand\glossentry[2]{%
18917 \glslongextraNameFmt{##1} &
18918 \glslongextraDescFmt{##1} &
18919 \glslongextraSymbolFmt{##1}\tabularnewline
18920 }%
18921 \renewcommand\subglossentry[3]{%
18922 \glslongextraSubNameFmt{##1}{##2} &
18923 \glslongextraSubDescFmt{##1}{##2} &
18924 \glslongextraSubSymbolFmt{##1}{##2}%
18925 \tabularnewline
18926 }%
18927 \ifglsnogroupskip
18928 \renewcommand*\glsgroupskip{}%
18929 \else
18930 \renewcommand*\glsgroupskip{\glspenaltygroupskip}%
18931 \fi
18932 }

```

mLocationHeader

```

18933 \newcommand\glslongextraNameDescSymLocationHeader{%
18934 \glslongextraNameDescSymLocationTabularHeader\endhead
18935 \glslongextraNameDescSymLocationTabularFooter\endfoot
18936 }

```

onTabularHeader

```

18937 \newcommand\glslongextraNameDescSymLocationTabularHeader{%
18938 \toprule
18939 \glslongextraHeaderFmt\entryname &
18940 \glslongextraHeaderFmt\descriptionname &
18941 \glslongextraHeaderFmt\symbolname &
18942 \glslongextraHeaderFmt\pagelistname\tabularnewline
18943 \midrule
18944 }

```

onTabularFooter

```

18945 \newcommand\glslongextraNameDescSymLocationTabularFooter{%
18946 \bottomrule

```

18947 }

me-desc-sym-loc Four columns: name, description and location

```
18948 \newglossarystyle{long-name-desc-sym-loc}%
18949 {%
18950   \ifGlsLongExtraUseTabular
18951     \renewenvironment{theglossary}%
18952       {%
18953         \glslongextraSymLocSetDescWidth
18954         \edef\@glslongextra@begintab{%
18955           \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
18956             \expandonce\glslongextraNameAlign
18957             \expandonce\glslongextraDescAlign
18958             \expandonce\glslongextraSymbolAlign
18959             \expandonce\glslongextraLocationAlign
18960           }}%
18961         \@glslongextra@begintab
18962       }%
18963       {%
18964         \glslongextraNameDescSymLocationTabularFooter
18965         \end{tabular}%
18966       }%
18967     \renewcommand*\glossaryheader{\glslongextraNameDescSymLocationTabularHeader}%
18968   \else
18969     \renewenvironment{theglossary}%
18970       {%
18971         \glspatchLToutput
18972         \glslongextraSymLocSetDescWidth
18973         \edef\@glslongextra@begintab{%
18974           \noexpand\begin{longtable}{%
18975             \expandonce\glslongextraNameAlign
18976             \expandonce\glslongextraDescAlign
18977             \expandonce\glslongextraSymbolAlign
18978             \expandonce\glslongextraLocationAlign
18979           }}%
18980         \@glslongextra@begintab
18981       }%
18982       {\end{longtable}}%
18983     \renewcommand*\glossaryheader{\glslongextraNameDescSymLocationHeader}%
18984   \fi
18985   \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{4}{##1}}%
18986   \renewcommand{\glossentry}[2]{%
18987     \glslongextraNameFmt{##1} &
18988     \glslongextraDescFmt{##1} &
18989     \glslongextraSymbolFmt{##1}&
18990     \glslongextraLocationFmt{##1}{##2}\tabularnewline
18991   }%
18992   \renewcommand{\subglossentry}[3]{%
18993     \glslongextraSubNameFmt{##1}{##2} &
```

```

18994 \glslongextraSubDescFmt{##1}{##2} &
18995 \glslongextraSubSymbolFmt{##1}{##2}&
18996 \glslongextraSubLocationFmt{##1}{##2}{##3}%
18997 \tabularnewline
18998 }%
18999 \ifglsgnogroupskip
19000 \renewcommand*{\glsgroupskip}{}%
19001 \else
19002 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
19003 \fi
19004 }

```

meSymDescHeader

```

19005 \newcommand{\glslongextraNameSymDescHeader}{%
19006 \glslongextraNameSymDescTabularHeader\endhead
19007 \glslongextraNameSymDescTabularFooter\endfoot
19008 }

```

scTabularHeader

```

19009 \newcommand{\glslongextraNameSymDescTabularHeader}{%
19010 \toprule
19011 \glslongextraHeaderFmt\entryname &
19012 \glslongextraHeaderFmt\symbolname &
19013 \glslongextraHeaderFmt\descriptionname\tabularnewline
19014 \midrule
19015 }

```

scTabularFooter

```

19016 \newcommand{\glslongextraNameSymDescTabularFooter}{%
19017 \bottomrule
19018 }

```

g-name-sym-desc Three column style with symbol in the second column.

```

19019 \newglossarystyle{long-name-sym-desc}%
19020 {%
19021 \ifGlsLongExtraUseTabular
19022 \renewenvironment{theglossary}%
19023 {%
19024 \glslongextraSymSetDescWidth
19025 \edef\@glslongextra@begintab{%
19026 \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
19027 \expandonce\glslongextraNameAlign
19028 \expandonce\glslongextraSymbolAlign
19029 \expandonce\glslongextraDescAlign
19030 }}%
19031 \@glslongextra@begintab
19032 }%
19033 {%
19034 \glslongextraNameSymDescTabularFooter

```

```

19035     \end{tabular}%
19036   }%
19037   \renewcommand*{\glossaryheader}{\glslongextraNameSymDescTabularHeader}%
19038 \else
19039   \renewenvironment{theglossary}%
19040   {%
19041     \glspatchLToutput
19042     \glslongextraSymSetDescWidth
19043     \edef\@glslongextra@begintab{%
19044       \noexpand\begin{longtable}{%
19045         \expandonce\glslongextraNameAlign
19046         \expandonce\glslongextraSymbolAlign
19047         \expandonce\glslongextraDescAlign
19048       }}%
19049     \@glslongextra@begintab
19050   }%
19051   {\end{longtable}}%
19052   \renewcommand*{\glossaryheader}{\glslongextraNameSymDescHeader}%
19053 \fi
19054 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{3}{##1}}%
19055 \renewcommand{\glossentry}[2]{%
19056   \glslongextraNameFmt{##1} &
19057   \glslongextraSymbolFmt{##1} &
19058   \glslongextraDescFmt{##1}\tabularnewline
19059 }%
19060 \renewcommand{\subglossentry}[3]{%
19061   \glslongextraSubNameFmt{##1}{##2} &
19062   \glslongextraSubSymbolFmt{##1}{##2} &
19063   \glslongextraSubDescFmt{##1}{##2}\tabularnewline
19064 }%
19065 \ifglsgroupskip
19066   \renewcommand*{\glsgroupskip}{}%
19067 \else
19068   \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
19069 \fi
19070 }

```

cLocationHeader

```

19071 \newcommand{\glslongextraNameSymDescLocationHeader}{%
19072   \glslongextraNameSymDescLocationTabularHeader\endhead
19073   \glslongextraNameSymDescLocationTabularFooter\endfoot
19074 }

```

onTabularHeader

```

19075 \newcommand{\glslongextraNameSymDescLocationTabularHeader}{%
19076   \toprule
19077   \glslongextraHeaderFmt\entryname &
19078   \glslongextraHeaderFmt\symbolname &
19079   \glslongextraHeaderFmt\descriptionname &

```

```

19080 \glslongextraHeaderFmt\pagelistname\tabularnewline
19081 \midrule
19082 }

```

onTabularFooter

```

19083 \newcommand{\glslongextraNameSymDescLocationTabularFooter}{%
19084 \bottomrule
19085 }

```

me-sym-desc-loc Four column style with symbol in the second column.

```

19086 \newglossarystyle{long-name-sym-desc-loc}%
19087 {%
19088 \ifGlsLongExtraUseTabular
19089 \renewenvironment{theglossary}%
19090 {%
19091 \glslongextraSymLocSetDescWidth
19092 \edef\@glslongextra@begintab{%
19093 \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
19094 \expandonce\glslongextraNameAlign
19095 \expandonce\glslongextraSymbolAlign
19096 \expandonce\glslongextraDescAlign
19097 \expandonce\glslongextraLocationAlign
19098 }}%
19099 \@glslongextra@begintab
19100 }%
19101 {%
19102 \glslongextraNameSymDescLocationTabularFooter
19103 \end{tabular}}%
19104 }%
19105 \renewcommand*\glossaryheader{\glslongextraNameSymDescLocationTabularHeader}%
19106 \else
19107 \renewenvironment{theglossary}%
19108 {%
19109 \glspatchLToutput
19110 \glslongextraSymLocSetDescWidth
19111 \edef\@glslongextra@begintab{%
19112 \noexpand\begin{longtable}{%
19113 \expandonce\glslongextraNameAlign
19114 \expandonce\glslongextraSymbolAlign
19115 \expandonce\glslongextraDescAlign
19116 \expandonce\glslongextraLocationAlign
19117 }}%
19118 \@glslongextra@begintab
19119 }%
19120 {\end{longtable}}%
19121 \renewcommand*\glossaryheader{\glslongextraNameSymDescLocationHeader}%
19122 \fi
19123 \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{4}{##1}}%
19124 \renewcommand{\glossentry}[2]{%

```

```

19125 \glslongextraNameFmt{##1} &
19126 \glslongextraSymbolFmt{##1} &
19127 \glslongextraDescFmt{##1} &
19128 \glslongextraLocationFmt{##1}{##2}\tabularnewline
19129 }%
19130 \renewcommand{\subglossentry}[3]{%
19131 \glslongextraSubNameFmt{##1}{##2} &
19132 \glslongextraSubSymbolFmt{##1}{##2} &
19133 \glslongextraSubDescFmt{##1}{##2} &
19134 \glslongextraSubLocationFmt{##1}{##2}{##3}\tabularnewline
19135 }%
19136 \ifglsgnognogroupskip
19137 \renewcommand*{\glsgroupskip}{}%
19138 \else
19139 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
19140 \fi
19141 }

```

meDescNameHeader

```

19142 \newcommand{\glslongextraSymDescNameHeader}{%
19143 \glslongextraSymDescNameTabularHeader\endhead
19144 \glslongextraSymDescNameTabularFooter\endfoot
19145 }

```

meTabularHeader

```

19146 \newcommand{\glslongextraSymDescNameTabularHeader}{%
19147 \toprule
19148 \glslongextraHeaderFmt\symbolname &
19149 \glslongextraHeaderFmt\descriptionname &
19150 \glslongextraHeaderFmt\entryname\tabularnewline
19151 \midrule
19152 }

```

meTabularFooter

```

19153 \newcommand{\glslongextraSymDescNameTabularFooter}{%
19154 \bottomrule
19155 }

```

g-sym-desc-name Three column style with symbol in the first column, description in the second and name in the third.

```

19156 \newglossarystyle{long-sym-desc-name}%
19157 {%
19158 \ifGlsLongExtraUseTabular
19159 \renewenvironment{theglossary}%
19160 {%
19161 \glslongextraSymSetDescWidth
19162 \edef\@glslongextra@begintab{%
19163 \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
19164 \expandonce\glslongextraSymbolAlign

```

```

19165         \expandonce\glslongextraDescAlign
19166         \expandonce\glslongextraNameAlign
19167     } }%
19168     \@glslongextra@begintab
19169 }%
19170 {%
19171     \glslongextraSymDescNameTabularFooter
19172     \end{tabular}%
19173 }%
19174 \renewcommand*{\glossaryheader}{\glslongextraSymDescNameTabularHeader}%
19175 \else
19176 \renewenvironment{theglossary}%
19177 {%
19178     \glspatchLToutput
19179     \glslongextraSymSetDescWidth
19180     \edef\@glslongextra@begintab{%
19181         \noexpand\begin{longtable}{%
19182             \expandonce\glslongextraSymbolAlign
19183             \expandonce\glslongextraDescAlign
19184             \expandonce\glslongextraNameAlign
19185         } }%
19186     \@glslongextra@begintab
19187 }%
19188 {\end{longtable}}%
19189 \renewcommand*{\glossaryheader}{\glslongextraSymDescNameHeader}%
19190 \fi
19191 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{3}{##1}}%
19192 \renewcommand{\glossentry}[2]{%
19193     \glslongextraSymbolFmt{##1} &
19194     \glslongextraDescFmt{##1} &
19195     \glslongextraNameFmt{##1}\tabularnewline
19196 }%
19197 \renewcommand{\subglossentry}[3]{%
19198     \glslongextraSubSymbolFmt{##1}{##2} &
19199     \glslongextraSubDescFmt{##1}{##2} &
19200     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
19201 }%
19202 \ifglsgroupskip
19203     \renewcommand*{\glsgroupskip}{}%
19204 \else
19205     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
19206 \fi
19207 }

```

mDescNameHeader

```

19208 \newcommand{\glslongextraLocationSymDescNameHeader}{%
19209 \glslongextraLocationSymDescNameTabularHeader\endhead
19210 \glslongextraLocationSymDescNameTabularFooter\endfoot
19211 }

```


meTabularHeader

```

19212 \newcommand{\glslongextraLocationSymDescNameTabularHeader}{%
19213   \toprule
19214   \glslongextraHeaderFmt\pagelistname &
19215   \glslongextraHeaderFmt\symbolname &
19216   \glslongextraHeaderFmt\descriptionname &
19217   \glslongextraHeaderFmt\entryname\tabularnewline
19218   \midrule
19219 }

```

meTabularFooter

```

19220 \newcommand{\glslongextraLocationSymDescNameTabularFooter}{%
19221   \bottomrule
19222 }

```

c-sym-desc-name Four column style with location list, symbol, description and name.

```

19223 \newglossarystyle{long-loc-sym-desc-name}%
19224 {%
19225   \ifGlsLongExtraUseTabular
19226     \renewenvironment{theglossary}%
19227     {%
19228       \glslongextraSymLocSetDescWidth
19229       \edef\@glslongextra@begintab{%
19230         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
19231           \expandonce\glslongextraLocationAlign
19232           \expandonce\glslongextraSymbolAlign
19233           \expandonce\glslongextraDescAlign
19234           \expandonce\glslongextraNameAlign
19235         }}%
19236       \@glslongextra@begintab
19237     }%
19238     {%
19239       \glslongextraLocationSymDescNameTabularFooter
19240       \end{tabular}%
19241     }%
19242     \renewcommand*\glossaryheader{\glslongextraLocationSymDescNameTabularHeader}%
19243   \else
19244     \renewenvironment{theglossary}%
19245     {%
19246       \glspatchLToutput
19247       \glslongextraSymLocSetDescWidth
19248       \edef\@glslongextra@begintab{%
19249         \noexpand\begin{longtable}{%
19250           \expandonce\glslongextraLocationAlign
19251           \expandonce\glslongextraSymbolAlign
19252           \expandonce\glslongextraDescAlign
19253           \expandonce\glslongextraNameAlign
19254         }}%
19255       \@glslongextra@begintab

```

```

19256 }%
19257 {\end{longtable}}%
19258 \renewcommand*{\glossaryheader}{\glslongextraLocationSymDescNameHeader}%
19259 \fi
19260 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{4}{##1}}%
19261 \renewcommand{\glossentry}[2]{%
19262   \glslongextraLocationFmt{##1}{##2} &
19263   \glslongextraSymbolFmt{##1} &
19264   \glslongextraDescFmt{##1} &
19265   \glslongextraNameFmt{##1}\tabularnewline
19266 }%
19267 \renewcommand{\subglossentry}[3]{%
19268   \glslongextraSubLocationFmt{##1}{##2}{##3} &
19269   \glslongextraSubSymbolFmt{##1}{##2} &
19270   \glslongextraSubDescFmt{##1}{##2} &
19271   \glslongextraSubNameFmt{##1}{##2}\tabularnewline
19272 }%
19273 \ifglsgnogroupskip
19274   \renewcommand*{\glsgroupskip}{}%
19275 \else
19276   \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
19277 \fi
19278 }

```

scSymNameHeader

```

19279 \newcommand{\glslongextraDescSymNameHeader}{%
19280   \glslongextraDescSymNameTabularHeader\endhead
19281   \glslongextraDescSymNameTabularFooter\endfoot
19282 }

```

meTabularHeader

```

19283 \newcommand{\glslongextraDescSymNameTabularHeader}{%
19284   \toprule
19285   \glslongextraHeaderFmt\descriptionname &
19286   \glslongextraHeaderFmt\symbolname &
19287   \glslongextraHeaderFmt\entryname\tabularnewline
19288   \midrule
19289 }

```

meTabularFooter

```

19290 \newcommand{\glslongextraDescSymNameTabularFooter}{%
19291   \bottomrule
19292 }

```

g-desc-sym-name Three column style with description in the first column, symbol in the second and name in the third.

```

19293 \newglossarystyle{long-desc-sym-name}%
19294 {%
19295   \ifGlsLongExtraUseTabular

```

```

19296 \renewenvironment{theglossary}%
19297 {%
19298     \glslongextraSymSetDescWidth
19299     \edef\@glslongextra@begintab{%
19300         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
19301             \expandonce\glslongextraDescAlign
19302             \expandonce\glslongextraSymbolAlign
19303             \expandonce\glslongextraNameAlign
19304         }}%
19305     \@glslongextra@begintab
19306 }%
19307 {%
19308     \glslongextraDescSymNameTabularFooter
19309     \end{tabular}%
19310 }%
19311 \renewcommand*\glossaryheader{\glslongextraDescSymNameTabularHeader}%
19312 \else
19313 \renewenvironment{theglossary}%
19314 {%
19315     \glspatchLToutput
19316     \glslongextraSymSetDescWidth
19317     \edef\@glslongextra@begintab{%
19318         \noexpand\begin{longtable}{%
19319             \expandonce\glslongextraDescAlign
19320             \expandonce\glslongextraSymbolAlign
19321             \expandonce\glslongextraNameAlign
19322         }}%
19323     \@glslongextra@begintab
19324 }%
19325 {\end{longtable}}%
19326 \renewcommand*\glossaryheader{\glslongextraDescSymNameHeader}%
19327 \fi
19328 \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{3}{##1}}%
19329 \renewcommand{\glossentry}[2]{%
19330     \glslongextraDescFmt{##1} &
19331     \glslongextraSymbolFmt{##1} &
19332     \glslongextraNameFmt{##1}\tabularnewline
19333 }%
19334 \renewcommand{\subglossentry}[3]{%
19335     \glslongextraSubDescFmt{##1}{##2} &
19336     \glslongextraSubSymbolFmt{##1}{##2} &
19337     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
19338 }%
19339 \ifglsgroupskip
19340     \renewcommand*\glsgroupskip{}%
19341 \else
19342     \renewcommand*\glsgroupskip{\glspenaltygroupskip}%
19343 \fi
19344 }

```

scSymNameHeader

```
19345 \newcommand{\glslongextraLocationDescSymNameHeader}{%
19346   \glslongextraLocationDescSymNameTabularHeader\endhead
19347   \glslongextraLocationDescSymNameTabularFooter\endfoot
19348 }
```

meTabularHeader

```
19349 \newcommand{\glslongextraLocationDescSymNameTabularHeader}{%
19350   \toprule
19351   \glslongextraHeaderFmt\pagelistname &
19352   \glslongextraHeaderFmt\descriptionname &
19353   \glslongextraHeaderFmt\symbolname &
19354   \glslongextraHeaderFmt\entryname\tabularnewline
19355   \midrule
19356 }
```

meTabularFooter

```
19357 \newcommand{\glslongextraLocationDescSymNameTabularFooter}{%
19358   \bottomrule
19359 }
```

c-desc-sym-name Four column style with location list, description, symbol and name.

```
19360 \newglossarystyle{long-loc-desc-sym-name}%
19361 {%
19362   \ifGlsLongExtraUseTabular
19363     \renewenvironment{theglossary}%
19364     {%
19365       \glslongextraSymLocSetDescWidth
19366       \edef\@glslongextra@begintab{%
19367         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
19368           \expandonce\glslongextraLocationAlign
19369           \expandonce\glslongextraDescAlign
19370           \expandonce\glslongextraSymbolAlign
19371           \expandonce\glslongextraNameAlign
19372         }}%
19373       \@glslongextra@begintab
19374     }%
19375     {%
19376       \glslongextraLocationDescSymNameTabularFooter
19377       \end{tabular}%
19378     }%
19379     \renewcommand*\glossaryheader{\glslongextraLocationDescSymNameTabularHeader}%
19380   \else
19381     \renewenvironment{theglossary}%
19382     {%
19383       \glspatchLToutput
19384       \glslongextraSymLocSetDescWidth
19385       \edef\@glslongextra@begintab{%
19386         \noexpand\begin{longtable}{%

```

```

19387         \expandonce\glslongextraLocationAlign
19388         \expandonce\glslongextraDescAlign
19389         \expandonce\glslongextraSymbolAlign
19390         \expandonce\glslongextraNameAlign
19391     } }%
19392     \@glslongextra@begintab
19393 }%
19394 {\end{longtable}}%
19395 \renewcommand*{\glossaryheader}{\glslongextraLocationDescSymNameHeader}%
19396 \fi
19397 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{4}{##1}}%
19398 \renewcommand{\glossentry}[2]{%
19399     \glslongextraLocationFmt{##1}{##2} &
19400     \glslongextraDescFmt{##1} &
19401     \glslongextraSymbolFmt{##1} &
19402     \glslongextraNameFmt{##1}\tabularnewline
19403 }%
19404 \renewcommand{\subglossentry}[3]{%
19405     \glslongextraSubLocationFmt{##1}{##2}{##3} &
19406     \glslongextraSubDescFmt{##1}{##2} &
19407     \glslongextraSubSymbolFmt{##1}{##2} &
19408     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
19409 }%
19410 \ifglsnogroupskip
19411     \renewcommand*{\glsgroupskip}{}%
19412 \else
19413     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
19414 \fi
19415 }

```

5 topic styles (glossary-topic.sty)

5.1 Package Initialisation and Options

Provides “topic” styles where top-level entries are considered a topic.

```
19416 \NeedsTeXFormat{LaTeX2e}
```

```
19417 \ProvidesPackage{glossary-topic}[2021/11/22 v1.48 (NLCT)]
```

Load required package.

```
19418 \RequirePackage{multicol}
```

The top-level entries act like headers. If the top-level entry has a description it’s placed below the name.

topic

```
19419 \newglossarystyle{topic}{%
```

```
19420   \renewenvironment{theglossary}{%
```

```
19421   {%
```

```
19422     \glstopicInit
```

```
19423     \def\glstopic@prechildren{}%
```

```
19424     \def\glstopic@prevlevel{-1}%
```

```
19425   }%
```

```
19426   {\par}%
```

```
19427   \renewcommand*{\glossaryheader}{}%
```

```
19428   \renewcommand*{\glsgroupheading}[1]{%
```

```
19429     \def\glstopic@prevlevel{-1}%
```

```
19430     \glstopicGroupHeading{##1}%
```

```
19431   }%
```

```
19432   \renewcommand{\glossentry}[2]{%
```

```
19433     \hangindentOpt\relax
```

```
19434     \parindent\glstopicParIndent\relax
```

```
19435     \glstopicItem{##1}{##2}%
```

If there isn’t a description, penalise a page break.

```
19436     \ifglshasdesc{##1}%
```

```
19437     {%
```

```
19438       \def\glstopic@prechildren{}%
```

```
19439     }%
```

```
19440     {%
```

```
19441       \def\glstopic@prechildren{\nopagebreak}%
```

```
19442     }%
```

```
19443   }%
```

```
19444   \renewcommand{\subglossentry}[3]{%
```

```
19445     \ifnum\glstopic@prevlevel=0\relax\glstopic@prechildren\fi
```

```
19446     \def\glstopic@prevlevel{##1}%
```

Grouping is added to scope the effect of `\everypar`.

```

19447 \begingroup
19448 \glstopicAssignSubIndent{##1}%
19449 \glstopicSubItem{##1}{##2}{##3}%

19450 \par
19451 \endgroup
19452 }%
19453 \renewcommand*{\glsgroupskip}{}%
19454 }

```

`\picGroupHeading`

`\glstopicGroupHeading{<group label>}`

May be redefined if letter group headings are required. For example:

```

\renewcommand*{\glstopicGroupHeading}[1]{%
  \glstrgetgrouptitle{#1}{\thisgrptitle}%
  \section*{\thisgrptitle}%
}

19455 \newcommand*{\glstopicGroupHeading}[1]{%

```

`\glstopicItem`

`\glstopicItem{<label>}{<location list>}`

```

19456 \newcommand*{\glstopicItem}[2]{%
19457 \glspare\glstopicPreSkip\glspare\noindent
19458 \glstopicMarker{#1}%
19459 \glstopicTitleFont
19460 {%
19461 \glstryitem{#1}\glstarget{#1}{\glstopicTitle{#1}}%
19462 }%
19463 \ifglshasdesc{#1}%
19464 {\glspare\nobreak\glstopicMidSkip\glspare\nobreak
19465 \@afterheading\glstopicDesc{#1}\glspare\glstopicPostSkip}%
19466 {\glspare\nobreak\glstopicPostSkip}%
19467 \glstopicLoc{#1}{#2}%
19468 }

```

`\glstopicMarker` May be used to insert a bookmark etc if required.

```

19469 \newcommand*{\glstopicMarker}[1]{%

```

`\glstopicName`

```

19470 \newcommand*{\glstopicTitle}[1]{\Glossentryname{#1}%
19471 \ifglshassymbol{#1}{\space\glossentrysymbol{#1}}{}}%
19472 }

```

stopicTitleFont

```
19473 \newcommand*{\glstopicTitleFont}[1]{\textbf{\large #1}}
```

\glstopicDesc

```
19474 \newcommand*{\glstopicDesc}[1]{\Glossentrydesc{#1}\glspostdescription}
```

\glstopicLoc

```
19475 \newcommand*{\glstopicLoc}[2]{}
```

stopicParIndent

```
19476 \newlength\glstopicParIndent
```

```
19477 \setlength\glstopicParIndent{20pt}
```

stopicSubIndent

```
19478 \newlength\glstopicSubIndent
```

```
19479 \setlength\glstopicSubIndent{20pt}
```

\glstopicInit

```
19480 \newcommand{\glstopicInit}{}%
```

AssignSubIndent

```
\glstopicAssignSubIndent{<level>}
```

Used to set the indentation for sub-levels.

```
19481 \newcommand*{\glstopicAssignSubIndent}[1]{%
```

```
19482   \par
```

```
19483   \parindent\dimexpr#1\glstopicSubIndent-\glstopicSubIndent\relax
```

```
19484   \glstopicAssignWidest{#1}%
```

```
19485   \glstopicsubitemhangindent=\dimexpr\parindent+\glstopicwidest\relax
```

```
19486   \hangindent\glstopicsubitemhangindent\relax
```

```
19487   \everypar{\hangindent\glstopicsubitemhangindent\relax
```

```
19488     \parindent\dimexpr\glstopicSubItemParIndent+\glstopicsubitemhangindent\relax}%
```

```
19489 }%
```

bitemhangindent

```
19490 \newlength\glstopicsubitemhangindent
```

ubItemParIndent

```
19491 \newlength\glstopicSubItemParIndent
```

```
19492 \glstopicSubItemParIndent\parindent
```

\glstopicwidest

```
19493 \newlength\glstopicwidest
```


picAssignWidest

`\glstopicAssignWidest{<level>}`

Used in the definition of `\glstopicAssignSubIndent` to set the indentation from the widest name for the given level. This will require `glossary-tree` to set the values.

```
19494 \newcommand*{\glstopicAssignWidest}[1]{%
19495   \ifcsundef{@glswidestlength\romannumeral#1}%
19496   {%
19497     \ifcsdef{@glswidestname\romannumeral#1}%
19498     {%
19499       \settowidth{\glstopicwidest}{%
19500         \glstopicSubNameFont{\csuse{@glswidestname\romannumeral#1}}%
19501         \glstopicSubItemSep
19502       }%
19503     }%
19504     {\setlength{\glstopicwidest}{0pt}}%
```

Save the value so that it doesn't have to keep being recalculated.

```
19505   \csedef{@glswidestlength\romannumeral#1}{\the\glstopicwidest}%
19506   }%
19507   {\setlength{\glstopicwidest}{\csuse{@glswidestlength\romannumeral#1}}}%
19508 }
```

glstopicPreSkip

```
19509 \newcommand*{\glstopicPreSkip}{\medskip}
```

glstopicMidSkip

```
19510 \newcommand*{\glstopicMidSkip}{\smallskip}
```

lstopicPostSkip

```
19511 \newcommand*{\glstopicPostSkip}{\smallskip}
```

glstopicSubItem

`\glstopicSubItem{<level>}{<label>}{<location list>}`

```
19512 \newcommand*{\glstopicSubItem}[3]{%
19513   \glstopicSubItemBox{#1}{\glstopicSubNameFont{\glentryitem{#2}}%
19514     \glstarget{#2}{\glossentryname{#2}}}%
19515   \glstopicSubItemSep
19516   }%
19517   \ifglshassymbol{#2}{(\glossentrysymbol{#2})\space}{}%
19518   \ifglshasdesc{#2}%
19519   {\glossentrydesc{#2}\glspostdescription\glstopicSubPreLocSep}{}%
19520   \glstopicSubLoc{#2}{#3}%
19521 }
```

topicSubItemSep

```
19522 \newcommand*{\glstopicSubItemSep}{\quad}
```

topicSubItemBox

```
\glstopicSubItemBox{<level>}{<text>}
```

```
19523 \newcommand*{\glstopicSubItemBox}[2]{%
```

```
19524   \ifdim\glstopicwidest>0pt\relax\makebox[\glstopicwidest][l]{#2}\else#2\fi
```

```
19525 }
```

opicSubNameFont

```
19526 \newcommand*{\glstopicSubNameFont}[1]{\textbf{#1}}
```

picSubPreLocSep

```
19527 \newcommand*{\glstopicSubPreLocSep}{\space}
```

\glstopicSubLoc

```
19528 \newcommand*{\glstopicSubLoc}[2]{#2}
```

\glstopicCols

```
19529 \newcommand*{\glstopicCols}{2}
```

glstopicColsEnv

```
19530 \newcommand*{\glstopicColsEnv}{multicols}
```

topicmcols

```
19531 \newglossarystyle{topicmcols}{%
```

```
19532   \renewenvironment{theglossary}{%
```

```
19533   {%
```

```
19534     \glstopicInit
```

```
19535     \def\glstopic@prechildren{}%
```

```
19536     \def\glstopic@postchildren{}%
```

```
19537     \def\glstopic@prevlevel{-1}%
```

```
19538   }%
```

```
19539   {%
```

```
19540     \ifnum\glstopic@prevlevel>0\relax\glstopic@postchildren\fi
```

```
19541     \par
```

```
19542   }%
```

```
19543   \renewcommand*{\glossaryheader}{}%
```

```
19544   \renewcommand*{\glsgroupheading}[1]{%
```

```
19545     \ifnum\glstopic@prevlevel>0\relax\glstopic@postchildren\fi
```

```
19546     \def\glstopic@prevlevel{-1}%
```

```
19547     \glstopicGroupHeading{##1}%
```

```
19548   }%
```

```
19549   \renewcommand{\glossentry}[2]{%
```

```

19550 \ifnum\glstopic@prevlevel>0\relax\glstopic@postchildren\fi
19551 \def\glstopic@prevlevel{0}%
19552 \hangindentOpt\relax
19553 \parindent\glstopicParIndent\relax
19554 \glstopicItem{##1}{##2}%
19555 \ifnum\glstopicCols>1\relax
    If there isn't a description, penalise a page break.
19556 \ifglshasdesc{##1}%
19557 {%
19558 \edef\glstopic@prechildren{%
19559 \noexpand\begin{\glstopicColsEnv}{\glstopicCols}%
19560 }%
19561 }%
19562 {%
19563 \edef\glstopic@prechildren{%
19564 \noexpand\nopagebreak
19565 \noexpand\begin{\glstopicColsEnv}{\glstopicCols}%
19566 }%
19567 }%
19568 \edef\glstopic@postchildren{\noexpand\end{\glstopicColsEnv}}%
19569 \fi
19570 }%
19571 \renewcommand{\subglossentry}[3]{%
19572 \ifnum\glstopic@prevlevel=0\relax\glstopic@prechildren\fi
19573 \def\glstopic@prevlevel{##1}%
19574 \glstopicAssignSubIndent{##1}%
19575 \glstopicSubItem{##1}{##2}{##3}%
19576 }%
19577 \renewcommand*{\glsgroupskip}{}%
19578 }

```

Glossary

bib2gls A command line Java application that selects entries from a .bib file and converts them to glossary definitions (like bibtex but also performs hierarchical sorting and collation, thus omitting the need for **xindy** or **makeindex**). Further details at: <http://www.dickimaw-books.com/software/bib2gls/..>

First use The first time a glossary entry is used (from the start of the document or after a reset) with one of the following commands: `\gls`, `\Gls`, `\GLS`, `\glspl`, `\Glspl`, `\GLSpl` or `\glsdisp`. *see* **First use flag** & **First use text**

First use flag A conditional that determines whether or not the entry has been used according to the rules of **first use**.

First use text The text that is displayed on **first use**, which is governed by the first and first-plural keys of `\newglossaryentry`. (May be overridden by `\glsdisp`.)

makeindex An indexing application.

xindy An flexible indexing application with multilingual support written in Perl.

Change History

0.1 (2015-11-22)		\@Glsymbol@: added redefinition	91
General: Initial experimental release	4	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	91
\Glsfmtshort: new	379	\@Glstext@: added redefinition	87
\glsfmtshort: new	378	\@Glsuseri@: added redefinition	92
\Glsfmtshortpl: new	379	\@Glsuserii@: added redefinition	92
\glsfmtshortpl: new	379	\@Glsuseriii@: added redefinition	92
short: switched inline full form to short		\@Glsuseriv@: added redefinition	93
(long)	265	\@Glsuseriv@: added redefinition	93
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	93
\@ACRlong: added redefinition	97	\@acrlong: added redefinition	96
\@ACRlongpl: added redefinition	98	\@acrlongpl: added redefinition	97
\@ACRshort: added redefinition	95	\@acrshort: added redefinition	94
\@ACRshortpl: added redefinition	96	\@acrshortpl: added redefinition	95
\@Acrlong: added redefinition	96	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	98	argument	77
\@Acrshort: added redefinition	94	\@glsdescplural@: added redefinition .	90
\@Acrshortpl: added redefinition	95	\@glsfirst@: added redefinition	87
\@GLSdesc@: added redefinition	90	\@glsfirstplural@: added redefinition	88
\@GLSdescplural@: added redefinition .	91	\@glsplural@: added redefinition	88
\@GLSfirst@: added redefinition	88	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	89	redefinition	91
\@GLSname@: added redefinition	90	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	88	new	157
\@GLSSymbol@: added redefinition	91	\@glsxtr@field@linkdefs: new	86
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	229
redefinition	92	\@print@glossary: added redefinition	154
\@GLStext@: added redefinition	87	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	92	\abbrvdefaultfont	235
\@GLSuserii@: added redefinition	92	\glsaccessdesc: new	187
\@GLSuseriii@: added redefinition	93	\glsaccessdescplural: new	188
\@GLSuseriv@: added redefinition	93	\glsaccessfirst: new	185
\@GLSuserv@: added redefinition	93	\glsaccessfirstplural: new	186
\@GLSuservi@: added redefinition	93	\Glsaccesslong: new	190
\@GLSdesc@: added redefinition	90	\glsaccesslong: new	190
\@GLSdescplural@: added redefinition .	90	\glsaccessname: new	183
\@Glsfirst@: added redefinition	87	\glsaccessplural: new	184
\@Glsfirstplural@: added redefinition	89	\Glsaccessshort: new	189
\@Glsname@: added redefinition	89	\glsaccessshort: new	189
\@Glsplural@: added redefinition	88	\Glsaccessshortpl: new	189

\glsaccesssshortpl: new	189	\cGLSpl: new	127
\glsaccessssymbol: new	186	\cGLSpl@: new	127
\glsaccessssymbolplural: new	187	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	184	new	122
\glstentryfmt: added check for short ..	77	\cGLS: new	126
\glslongpltok: new	229	\cGLSformat: new	126
\glsshortpltok: new	229	\cGLSpl: new	126
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	127
name in \setkeys	231	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	18
for plural	226	\glsenableentrycount: new	122
\GLSxtrlongpl: new	246	\glsfirstabbrvdefaultfont: new ..	235
\Glsxtrlongpl: new	245	\glsfirstlongdefaultfont: new ...	235
\glsxtrlongpl: new	245	\Glsfmtfirst: new	382
\glsxtrNoGlossaryWarning: new	24	\glsfmtfirst: new	382
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	383
new	226	\glsfmtfirstpl: new	383
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	381
new	226	\glsfmtplural: new	381
\glsxtrpostlinkendsentence: new ..	225	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	244	\Glsxtrtitleshort	379
\Glsxtrshortpl: new	243	renamed from \glstentryfmtshort ..	379
\glsxtrshortpl: new	243	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort	378
\glslabeltok	257	renamed from \glstentryfmtshort ..	378
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	255	\Glsxtrtitleshortpl	379
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl	379
redefinition of \acronymtype	20	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	379
\Glsxtrshort	379	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl	379
\glsxtrshort	378	\Glsfmttext: new	381
\Glsfmtshortpl: changed to use		\glsfmttext: new	380
\glsxtrshortpl	379	\glshasattribute: new	203
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	202
\glsxtrshortpl	379	\glsxtremsuffix: new	306
\glsxtrifemptyglossary: new	34	\GlsXtrEnableEntryCounting: new ..	121
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new	124
argument	206	\glsxtrscfont: new	272
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new	273
argument	206	\glsxtrsmfont: new	289
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new	289
default type to \acronymtype	136	short-em: new	314
\newterm: fixed name argument	205	short-em-desc: new	315
0.5 (2015-12-07)		short-em-footnote: new	325
\@cGLS: new	126	short-em-long: new	310
\@cGLS@: new	126	short-em-long-desc: new	311

short-em-postfootnote: new	328	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new	284	attribute	368
short-sc-postfootnote: new	286	\Glsxtrheadtext: now uses headuc	
short-sm: new	293	attribute	371
short-sm-desc: new	295	\glxtrheadtext: now uses headuc	
short-sm-footnote: new	300	attribute	371
short-sm-long: new	291	short-em-footnote: switch off regular	
short-sm-long-desc: new	293	attribute if set	326
short-sm-postfootnote: new	303	short-em-footnote-desc: switch off	
long-noshort-em: new	318	regular attribute if set	327
long-noshort-em-desc: new	322	short-long: switch off regular attribute	
long-noshort-sm: new	297	if set	256
long-noshort-sm-desc: new	299	short-long-desc: switch off regular	
long-short-em: new	306	attribute if set	257
long-short-em-desc: new	307	short-postfootnote-desc: switch off	
long-short-sm: new	290	regular attribute if set	263
long-short-sm-desc: new	291	short-sc-footnote: switch off regular	
0.5.1 (2015-12-02)		attribute if set	284
\Glsaccessstext: new	184	short-sc-footnote-desc: switch off	
0.5.1 (2015-12-07)		regular attribute if set	286
\@glxtr@doaccsupp: new	23	short-sm-footnote: switch off regular	
General: removed \ifglxtruseuchead	368	attribute if set	301
\Glsaccessdesc: new	188	short-sm-footnote-desc: switch off	
\Glsaccessdescplural: new	188	regular attribute if set	303
\Glsaccessfirst: new	185	long-short: switch off regular attribute	
\Glsaccessfirstplural: new	186	if set	253
\Glsaccessname: new	184	long-short-desc: switch off regular	
\Glsaccessplural: new	185	attribute if set	255
\Glsaccesssymbol: new	186	long-short-sc-desc: switch off regular	
\Glsaccesssymbolplural: new	187	attribute if set	274
\Glsxtrheadfirst: now uses headuc		footnote: switch off regular attribute if	
attribute	373	set	258
\glxtrheadfirst: now uses headuc		postfootnote: switch off regular	
attribute	373	attribute if set	261
\Glsxtrheadfirstplural: now uses		0.5.2 (2015-12-08)	
headuc attribute	374	\@GLSdesc@: added accessibility support	90
\glxtrheadfirstplural: now uses		\@GLSdescplural@: added accessibility	
headuc attribute	373	support	91
\Glsxtrheadplural: now uses headuc		\@GLSfirst@: added accessibility	
attribute	372	support	88
\glxtrheadplural: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	372	support	89
\Glsxtrheadshort: now uses headuc		\@GLSname@: added accessibility support	90
attribute	369	\@GLSplural@: added accessibility	
\glxtrheadshort: now uses headuc		support	88
attribute	368	\@GLSsymbol@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support	91
attribute	369	\@GLSsymbolplural@: added	
		accessibility support	92

\@GLStext@: added accessibility support	87	\GLSaccesslongpl: new	190, 200
\@Glsdesc@: added accessibility support	90	\Glsaccesslongpl: new	190
\@Glsdescplural@: added accessibility support	90	\glsaccesslongpl: new	190
\@Glsfirst@: added accessibility support	87	\GLSaccessname: new	184, 197
\@Glsfirstplural@: added accessibility support	89	\GLSaccessplural: new	185, 197
\@Glsname@: add accessibility support	89	\GLSaccessshort: new	189, 199
\@Glsplural@: added accessibility support	88	\GLSaccessshortpl: new	190, 199
\@Glsymbol@: added accessibility support	91	\GLSaccesssymbol: new	187, 198
\@Glsymbolplural@: added accessibility support	91	\GLSaccesssymbolplural: new	187, 198
\@Glstext@: added accessibility support	87	\GLSaccessstext: new	184, 197
\@Glsdesc@: added accessibility support	90	\glsentryfmt: moved	
\@Glsdescplural@: added accessibility support	90	\glssetabbrvfmt from	
\@Glsfirst@: added accessibility support	87	\glsxtrabbrvfmt to here	77
\@Glsfirstplural@: added accessibility support	88	\GlsXtrEnableInitialTagging: new	221
\@Glsname@: added accessibility support	89	\glsxtrfieldtitlecase: new	207
\@Glsplural@: added accessibility support	88	\GlsXtrFormatLocationList: new	75
\@Glsymbol@: added accessibility support	91	\glsxtrnewabbrevpresetkeyhook:	
\@Glsymbolplural@: added accessibility support	91	new	233
\@Glstext@: added accessibility support	87	\glsxtrtagfont: new	223
\@Glsdesc@: added accessibility support	90	\KV@printgloss@nonumberlist: added	76
\@Glsdescplural@: added accessibility support	90	\mfu@checkword@do: added	222
\@Glsfirst@: added accessibility support	87	\setabbreviationstyle: added check	
\@Glsfirstplural@: added accessibility support	88	for post-definition style switch	250
\@Glsname@: added accessibility support	89	0.5.3 (2015-12-09)	
\@Glsplural@: added accessibility support	88	\@glsxtr@autoindex@at: new	218
\@Glsymbol@: added accessibility support	91	\@glsxtr@autoindex@encap: new	218
\@Glsymbolplural@: added accessibility support	91	\@glsxtr@autoindex@esc: new	219
\@Glstext@: added accessibility support	86	\@glsxtr@autoindex@level: new	219
\@glsxtr@activate@initialtagging:		\@glsxtr@autoindex@setname: new	217
new	223	\@glsxtr@doabbreviationsdef: new	20
\@glsxtr@do@titlecaps@warn: new	223	General: removed	
\@glsxtr@tag: new	223	\GlsXtrNoGlsWarningNoAutoMakeMain	156
General: fixed typo in glossaries-accsupp		\glsdescwidth: added	74
and tidied up code to use just one		\glspagelistwidth: added	74
\@ifpackageloaded	183	\glsxtrdoautoindexname: new	216
removed \glsxtrabbrvfmt	247	\glsxtrpostnamehook: new	213
\glossaryentrynumbers: added	74	\if@glsxtr@format@override: new	215
\Glossentrydesc: added	220	\ProvidesGlossariesExtraLang: new	437
\Glossentryname: added	212	\RequireGlossariesExtraLang: new	437
\Glossentrysymbol: added	221	0.5.4 (2015-12-15)	
\glossentrysymbol: added	220	\@newglossaryentry@defunitcounters:	
\GLSaccessdesc: new	188, 199	new	127
\GLSaccessdescplural: new	188, 199	\@GLSxtr@p@acrlong@: new	113
\GLSaccessfirst: new	185, 198	\@GLSxtr@p@acrlongpl@: new	113
\GLSaccessfirstplural: new	186, 198	\@GLSxtr@p@acrshort@: new	112
\GLSaccesslong: new	190, 200	\@GLSxtr@p@acrshortpl@: new	113
		\@GLSxtr@p@long@: new	112
		\@GLSxtr@p@longpl@: new	112

\@GLSxtr@p@plural@: new	111	\@sGlsXtrEnableOnTheFly: new	70
\@GLSxtr@p@short@: new	111	\cGlsformat: added	127
\@GLSxtr@p@shortpl@: new	112	\cGlsformat: added	127
\@GLSxtr@p@text@: new	111	\cGlsplformat: added	127
\@GlsXtrEnableOnTheFly: new	70	\cGlsplformat: added	127
\@Glsxtr: new	71	\gl:disablehyper: added	109
\@Glsxtr@p@acrlong@: new	113	\gl:donohyperlink: added	109
\@Glsxtr@p@acrlongpl@: new	113	\gl:senableentryunitcount: new ...	130
\@Glsxtr@p@acrshort@: new	112	\gl:shasattribute: added check for	
\@Glsxtr@p@acrshortpl@: new	112	entry's existence	203
\@Glsxtr@p@long@: new	112	\gl:sifattribute: added check for	
\@Glsxtr@p@longpl@: new	112	entry's existence	203
\@Glsxtr@p@plural@: new	111	\gl:spostlinkhook: added existence	
\@Glsxtr@p@short@: new	111	check	224
\@Glsxtr@p@shortpl@: new	111	\Glsxtr: new	71
\@Glsxtr@p@text@: new	110	\gl:xtr: new	70
\@Glsxtrpl: new	72	\gl:xtrcat: new	70
\@alt@gl:shyp@opt: new	105	\gl:xtrdohyperlink: added	108
\@gl:alt@hyp@opt: new	105	\gl:xtrdowrglossaryhook: new	105
\@gl:alt@hyp@opt@char: new	105	\GlsXtrEnableEntryUnitCounting:	
\@gl:alt@hyp@opt@keys: new	105	new	133
\@gl:increment@currunitcount:		\GlsXtrEnableOnTheFly: new	69
new	128	\Glsxtrpl: new	71
\@gl:local@increment@currunitcount:		\gl:xtrpl: new	71
new	129	\gl:xtrpostlocalreset: new	121
\@gl:setdefault@gl:link@opts:		\gl:xtrpostlocalunset: new	120
new	102	\gl:xtrpostreset: new	120
\@gl:xtr: new	70	\gl:xtrpostunset: new	119
\@gl:xtr@addunitcounter: new	128	\gl:xtrprotectlinks: new	110
\@gl:xtr@currunitcount: new	129	\GlsXtrSetAltModifier: new	105
\@gl:xtr@ifunitcounter: new	128	\GlsXtrSetDefaultGlsOpts: new ...	104
\@gl:xtr@p@acrlong@: new	113	\gl:xtrstarflywarn: new	70
\@gl:xtr@p@acrlongpl@: new	113	\GlsXtrWarning: new	72
\@gl:xtr@p@acrshort@: new	112	\MakeAcronymsAbbreviations: now	
\@gl:xtr@p@acrshortpl@: new	112	disables \setacronymstyle	136
\@gl:xtr@p@long@: new	112	1.0 (2016-01-24)	
\@gl:xtr@p@longpl@: new	112	\@gl:xtr@autoindexcrossrefs: new .	17
\@gl:xtr@p@plural@: new	111	\@gl:xtr@idx@displaynumberlist:	
\@gl:xtr@p@short@: new	111	new	147
\@gl:xtr@p@shortpl@: new	111	\@gl:xtr@idx@entrynumberlist: new	149
\@gl:xtr@p@text@: new	110	\@gl:xtr@noidx@displaynumberlist:	
\@gl:xtr@prevunitcount: new	129	new	147
\@gl:xtr@setentryunitcountunsetattr:		\@gl:xtr@noidx@entrynumberlist:	
new	133	new	148
\@gl:xtr@unitcountlist: new	128	\@gl:xtr@noidx@numberlistloop:	
\@gl:xtrpl: new	71	new	148
\@newglossaryentryposthook: added		\@gl:xtr@reg@glosslist: new	138
empty see value if not set and added		\makeglossaries: new	139
'see' to field key map	57		

1.01 (2016-02-02)	
\glxtrdiscardperiod: added check for first use	226
short-desc: fixed typo in \glxtrininlinefullformat and added missing second argument ...	266
1.02 (2016-04-25)	
\@glxtr@current@style: new	73
\Glsfmtfull: new	386
\glsfmtfull: new	386
\Glsfmtfullpl: new	387
\glsfmtfullpl: new	386
\Glsfmtlong: new	384
\glsfmtlong: new	384
\Glsfmtlongpl: new	385
\glsfmtlongpl: new	385
\Glsxtrheadfull: new	377
\glxtrheadfull: new	376
\Glsxtrheadfullpl: new	378
\glxtrheadfullpl: new	377
\Glsxtrheadlong: new	375
\glxtrheadlong: new	374
\Glsxtrheadlongpl: new	376
\glxtrheadlongpl: new	375
\Glsxtrtitlefull: new	377
\glxtrtitlefull: new	377
\Glsxtrtitlefullpl: new	378
\glxtrtitlefullpl: new	377
\Glsxtrtitlelong: new	375
\glxtrtitlelong: new	375
\Glsxtrtitlelongpl: new	376
\glxtrtitlelongpl: new	375
short-postfootnote-desc: added redef of \glxtrsetupfulldefs ..	264
\ifglxtrininsertinside: new	253
postfootnote: added redef of \glxtrsetupfulldefs	262
stylemods: new	24
1.03 (2016-04-27)	
\@GLSfirstplural@: bug fix: misspelt cs name	89
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	88
\@Glsfirstplural@: bug fix: misspelt cs name	89
\@Glsplural@: fixed bug \@Glsplural@ should be redefined not \@Glsplural	88
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	88
\glxtrtitlelongpl: bug fix: changed \glxtrlong to \glxtrlongpl ..	375
\glxtrtitleshortpl: bug fix: changed \glxtrshort to \glxtrshortpl	369
1.04 (2015-04-30)	
short-em-footnote: renamed from “footnote-em”	325
1.04 (2016-05-02)	
\@@glxtrpostloctag: new	76
\@GLSdesc@: set abbreviation and regular format	90
\@GLSdescplural@: set abbreviation and regular format	91
\@GLSfirst@: set abbreviation format ..	88
\@GLSfirstplural@: set abbreviation and regular format	89
\@GLSname@: set abbreviation and regular format	90
\@GLSplural@: set abbreviation and regular format	88
\@GLSsymbol@: set regular format	91
\@GLSsymbolplural@: set regular format	92
\@GLStext@: set abbreviation and regular format	87
\@GLSuseri@: set regular format	92
\@GLSuserii@: set regular format	92
\@GLSuseriii@: set regular format	93
\@GLSuseriv@: set regular format	93
\@GLSuservi@: set regular format	93
\@GLSdesc@: set abbreviation and regular format	90
\@GLSdescplural@: set abbreviation and regular format	90
\@Glsfirst@: set abbreviation and regular format	87
\@Glsfirstplural@: set abbreviation and regular format	89
\@Glsname@: set abbreviation and regular format	89
\@Glsplural@: set abbreviation and regular format	88
\@Glsymbol@: set regular format	91
\@Glsymbolplural@: set regular format	91
\@Glstext@: set abbreviation and regular format	87
\@Glsuseri@: set regular format	92
\@Glsuserii@: set regular format	92
\@Glsuseriii@: set regular format	92

\@Glsuseriv@: set regular format	93	\glslongfont: new	235
\@Glsuserv@: set regular format	93	\glslonguserfont: new	331
\@Glsuservi@: set regular format	93	\glsxtrassignfieldfont: new	86
\@gls@preglossaryhook: added check for entry's existence	223	\GlsXtrEnablePreLocationTag: new	75
\@glsdesc@: set abbreviation and regular format	90	\glsxtrfirstscfont: new	273
\@glsdescplural@: set abbreviation and regular format	90	\glsxtrfirstsmfont: new	289
\@glsfirst@: set abbreviation and regular format	87	\glsxtrlongshortdescsort: new	254
\@glsfirstplural@: set abbreviation and regular format	88	\glsxtrpostnamehook: added category check	213
\@glsname@: set abbreviation and regular format	89	\glsxtrregularfont: new	77
\@glsplural@: set abbreviation and regular format	88	\glsxtruserfield: new	331
\@glsymbol@: set regular format	91	\glsxtruserparen: new	331
\@glsymbolplural@: set regular format	91	\glsxtrusersuffix: new	331
\@glstext@: set abbreviation and regular format	86	\GlsXtrWarnDeprecatedAbbrStyle: new	252
\@glsxtr@deprecated@abbrstyle: new	252	short-em-long-em: new	312
\@glsxtr@do@style: new	25	short-em-long-em-desc: new	313
\@glsxtr@dolocktag: new	76	short-em-nolong: new	315
\@glsxtr@idx@entrynumberlist: switched from \let to \newcommand	149	short-em-nolong-desc: new	317
\@glsxtr@pagetag: new	76	short-em-postfootnote: renamed from "postfootnote-em"	328
\@glsxtr@pagetag: new	76	short-footnote: new	260
\@glsxtr@prelocktag: new	76	short-long-user: new	342
\@glsxtr@postlocktag: new	76	short-long-user-desc: new	343
\@glsxtr@prelocktag: new	75, 76	short-nolong: new	266
\glossentrydesc: added glossdescfont attribute check	208	short-nolong-desc: new	267
\Glossentryname: added glossnamefont attribute check	212	short-postfootnote: new	263
\glossentryname: added glossnamefont attribute check	209	short-sc-footnote: renamed from "footnote-sc"	284
moved post name hook inside condition	212	short-sc-nolong: new	278
\glsabbrvmfont: new	306	short-sc-nolong-desc: new	280
\glsabbrvuserfont: new	331	short-sc-postfootnote: renamed from "postfootnote-sc"	286
\glsfirstabbrvmfont: new	306	short-sm-footnote: renamed from "footnote-sm"	300
\glsfirstabbrvuserfont: new	331	short-sm-nolong: new	295
\glsfirstlongemfont: new	306	short-sm-nolong-desc: new	296
\glsfirstlonguserfont: new	331	short-sm-postfootnote: renamed from "postfootnote-sm"	303
\glsifnotregularcategory: new	204	\letabbreviationstyle: new	251
\glslongdefaultfont: new	235	\newabbreviationstyle: bug fix: corrected test for existence	250
\glslongemfont: new	306	long-em-noshort-em: new	319
		long-em-noshort-em-desc: new	323
		long-em-short-em: new	308
		long-em-short-em-desc: new	309
		long-noshort: new	272
		long-noshort-desc: new	271

long-noshort-em: renamed from “long-em”	318	\glxtrComputeTreeIndent: new ...	517
long-noshort-em-desc: renamed from “long-desc-em”	322	\glxtrComputeTreeSubIndent: new	517
long-noshort-sc: renamed from “long-sc”	280	\glxtrtreetopindent: new	508
long-noshort-sc-desc: renamed from “long-desc-sc”	282	short-em-long: fixed incorrect font used by long form	311
long-noshort-sm: renamed from “long-sm”	297	\xglsssetwidest: new	509
long-noshort-sm-desc: renamed from \long-desc-sm	299	1.06 (2016-06-18)	
long-short-user: new	332	\@glstdoifexistsorwarn: new	17
long-short-user-desc: new	341	\@glxtr@docdefval: new	16
\renewabbreviationstyle: new	251	\@glxtr@usesee: new	58
style: new	25	General: disabled docdef key at the start of the document	33
1.05 (2016-06-10)		docdef option changed to choice	16
\eglssetwidest: new	509	\glxtr@usesee: new	58
\glsFindWidestAnyName: new	511	\glxtrusesee: new	57
\glsFindWidestAnyNameLocation: new	517	\glxtruseseeformat: new	58
\glsFindWidestAnyNameSymbol: new	514	\if@glxtrdocdefrestricted: new ..	17
\glsFindWidestAnyNameSymbolLocation: new	515	1.07 (2016-08-15)	
\glsFindWidestLevelTwo: new	513	\@@glxtrp: new	113
\glsFindWidestUsedAnyName: new ..	511	\@GLSfirst@: added check for nohyperfirst attribute	88
\glsFindWidestUsedAnyNameLocation: new	516	\@GLSfirstplural@: added check for nohyperfirst attribute	89
\glsFindWidestUsedAnyNameSymbol: new	514	\@GLSxtrp: new	114
\glsFindWidestUsedAnyNameSymbolLocation: new	515	\@GLSfirst@: added check for nohyperfirst attribute	87
\glsFindWidestUsedLevelTwo: new .	512	\@GLSfirstplural@: added check for nohyperfirst attribute	89
\glsFindWidestUsedTopLevelName: new	510	\@GLSxtrp: new	114
\glsfirstlongfootnotefont: new ..	258	\@gls@preglossaryhook: added \glossxtrsetpopts	224
\glsgetwidestname: new	510	\@GLSfirst@: added check for nohyperfirst attribute	87
\glsgetwidestsubname: new	510	\@GLSfirstplural@: added check for nohyperfirst attribute	89
\glslongfootnotefont: new	258	\@glxtrinmark: new	366
\glxtrAltTreeIndent: new	508	\@glxtrnotinmark: new	366
\glxtralttreeInit: new	508	\@glxtrp: new	114
\glxtrAltTreePar: new	508	\@glxtrp@opt: new	113
\glxtrAltTreeSetHangIndent: new	518	\glossxtrsetpopts: new	113
\glxtrAltTreeSetSubHangIndent: new	518	\glsp: new	116
\glxtralttreeSubSymbolDescLocation: new	508	\glsp: new	116
\glxtralttreeSymbolDescLocation: new	507	\glxtr@entry@p: new	115
		\glxtrabbrvfootnote: new	258
		\glxtrchecknohyperfirst: new	87
		\glxtrfieldtitlecasecs: new	207
		\glxtrifinmark: new	366
		\GLSxtrp: new	117
		\Glsxtrp: new	116

\glxtrp: new	115	\glxtrassignfieldfont: added check for existence	86
\glxtrsetpopts: new	113	\glxtrresourcefile: new	158
short-long-desc: added text key	257	\printunsrtglossaries: new	165
fixed misspelling of \glsabbrvfont in plural key	257	\printunsrtglossary: new	165
long-short-desc: added missing text key	255	1.09 (2016-12-16)	
fixed misspelling of \glsabbrvfont .	255	\@glxtr@gettype: new	147
footnote: changed first forms to use \glsfirstlongfootnotefont ...	258	\@glxtr@mixed@assign@sortkey: new	147
postfootnote: removed \footnote from first keys	261	\@printglossary: redefined to save options	145
switched from \glsfirstlongfont to \glsfirstlongfootnotefont ...	262	\glxtr@makeglossaries: new	147
\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glxtrifwasfirstuse	137	1.10 (2016-12-17)	
1.08 (2016-12-13)		\@GLSpl@: fixed bug caused by typo in command name	79
\@glxtr@record: new	7	1.11 (2017-01-19)	
\@GLS@: added \@glxtr@record	79	\@glxtr@do@redef@forglsentries: new	5
\@GLSpl@: added \@glxtr@record ...	79	\@glxtr@noidx@do: new	172
\@Gls@: added \@glxtr@record	78	\@glxtr@redef@forglsentries: new .	5
\@GLspl@: added \@glxtr@record ...	79	\@glxtr@shortcutsval: new	22
\@gls@: added \@glxtr@record	78	\@glxtr@unsrt@getgrouptitle: new	171
\@gls@link@: added \@glxtr@record	79	\@print@noidx@glossary: added redefinition	150
\@gls@field@link: added \@glxtr@record	78	\glxtr@addloclistfield: added group key	13
\@gls@saveentrycounter: new	32	added location key	13
\@glsdisp: added \@glxtr@record ..	79	\glxtr@fields: new	160
\@glspl@: added \@glxtr@record ...	78	\glxtr@linkprefix: new	160
\@glxtr@dorecord: new	9	\glxtr@org@newignoredglossary: new	53
\@glxtr@err@undefaction: new	5	\glxtr@s@newignoredglossary: new	53
\@glxtr@record: new	6	\glxtr@shortcutsval: new	160
\@glxtr@warn@onexistsordo: new ...	5	\glxtr@texencoding: new	160
\@glxtr@warn@undefaction: new	5	\glxtr@writefields: new	160
\@print@unsrt@glossary: new	166	\GlsXtrLoadResources: new	160
record: added record package option ...	15	\glxtrpageref: new	49
\glsadd: added \@glxtr@record	85	\glxtrresourcefile: changed extension to .glstex	158
\glsdoifexists: now defines \glslabel	56	\newignoredglossary: added starred version	52
\glxtr@do@wrglossary: new	31	1.12 (2017-02-03)	
\glxtr@addloclistfield: new	12	\@glxtr@recordcounter: new	12
\glxtr@indexonly@saveentrycounter: new	12	\@gls@preglossaryhook: check for definition	223
\glxtr@record: new	162	\@glxtr@counterrecordhook: new .	162
\glxtr@resource: new	160	\@glxtr@display@loc: new	151
\glxtr@saveentrycounter: new	32	\@glxtr@docounterrecord: new ...	163
\glxtr@setup@record: new	12		

\@glxstr@longnewglossaryentry: new	52	\glxstrfieldifinlist:new	38
\@glxstr@noop@recordcounter:new ..	12	\glxstrfieldlistadd:new	37
\@glxstr@op@recordcounter:new ...	12	\glxstrfieldlisteadadd:new	37
\@glxstr@provide@storagekey:new ..	34	\glxstrfieldlistgadd:new	37
\@glxstr@s@longnewglossaryentry: new	51	\glxstrfieldlistxadd:new	37
\@glxstr@entryfmt:new	37	\glxstrfieldxifinlist:new	38
\@glxstr@indexaliases:new	103	\glxstrfmt:new	35
\@glxstr@setaliasnoindex:new	103	\GlsXtrFmtDefaultOptions:new	35
\@newglossaryentryposthook: added check for alias key	65	\GlsXtrFmtField:new	35
\@no@glxstr@indexaliases:new	103	\glxstrifkeydefined:new	34
\@printunsrtglossary:new	165	\glxstrindexaliases:new	104
General: added target key to printgloss family	145	\GlsXtrLetField:new	45
\apptoglossarypreamble:new	50	\GlsXtrLetFieldToField:new	45
\csGlsXtrLetField:new	45	\GlsXtrLoadResources: removed restriction on only one per document	160
\eGlsXtrSetField:new	45	\glxstrlocrangefmt:new	152
\gGlsXtrSetField:new	45	\glxstrpostlongdescription:new ..	52
\glxnoidxdisplayloc: added redefinition	151	\glxstrprovidestoragekey:new	34
\glssettocitle: added patch	54	\GlsXtrRecordCounter:new	162
\glxstr@counterrecord:new	162	\glxstrresourcecount:new	160
\glxstr@langtag:new	160	\glxstrresourcefile: added catcode change for @	159
\glxstr@newabbreviation:new	230	\glxstrsetaliasnoindex:new	103
\glxstr@org@newignoredglossary: Added check for existence	53	\GlsXtrSetField:new	45
\glxstr@pluralsuffixes:new	160	\glxstrsetfieldifexists:new	45
\glxstr@provideignoredglossary: new	54	\glxstrunsrtdo:new	171
\glxstr@s@newignoredglossary: Added check for existence	53	\GlsXtrusefield:new	43
\glxstr@s@provideignoredglossary: new	55	\glxstrusefield:new	43
\glxstr@abbrvpluralsuffix:new ...	235	short-postlong-user:new	338
\glxstr@alias:new	65	short-postlong-user-desc:new ...	340
\glxstr@copytoglossary:new	55	\longnewglossaryentry: added starred version	51
\glxstr@deffield:new	44	long-postshort-user:new	333
\glxstr@displayendloc:new	152	long-postshort-user-desc:new ...	337
\glxstr@displayendlochook:new ...	152	postdot:new	18
\glxstr@displayingleloc:new	152	\pretoglossarypreamble:new	50
\glxstr@displaystartloc:new	152	\print@noop@unsrtglossaryunit: new	171
\glxstr@dohyperlink: added check for alias field	108	\print@op@unsrtglossaryunit:new	170
\glxstr@deffield:new	44	\printunsrtglossary: added starred form	165
\glxstr@entryfmt:new	36	\printunsrtglossaryhandler:new ..	170
\glxstr@fieldddolistloop:new	38	\printunsrtglossaryunit:new	12
\glxstr@fieldforlistloop:new	38	\printunsrtglossaryunitsetup:new	171
		\provideignoredglossary:new	54
		\s@glxstr@provide@storagekey:new	35
		\s@printunsrtglossary:new	165
		\xGlsXtrSetField:new	45

1.13 (2017-02-07)		long-em-short-em: fixed spelling of	
\@glsdisp: removed		\glsabbrvfont	309
\@glsxtr@org@glsdisp	79	long-postshort-user: fixed spelling of	
\glsxtrsetaliasnoindex: switched to		\glsabbrvfont	333
\providecommand	103	long-postshort-user-desc: fixed	
1.14 (2017-04-18)		spelling of \glsabbrvfont	337
\@gls@link: added redefinition	82	long-short: fixed spelling of	
\@gls@noidx@getgrouptitle: new	149	\glsabbrvfont	253
\@gls@removespaces: new	153	long-short-user: fixed spelling of	
\@glsxtr@do@automake@err: new	162	\glsabbrvfont	332
\@glsxtr@org@gloautosee: new	30	footnote: fixed spelling of	
\@glsxtr@record: added third arg	6	\glsabbrvfont	258
\@glsxtr@recordsee: new	12	postfootnote: fixed spelling of	
General: added \glsadd option		\glsabbrvfont	261
theHvalue	85	1.16 (2017-06-15)	
added \glsadd option thevalue	85	\@glo@autosee: added redefinition	30
\glsdisablehyper: added redefinition	109	\@gls@noidx@getgrouptitle: fixed	
\glsenableentrycount: fixed		bug	149
assignment of \@cGls@	123	\@glsxtr@addunusedxrefs: added	
\glsenableentryunitcount: fixed		check for seealso field	66
assignment of \@cGls@	131	\@glsxtr@checkgroup: use \csuse	
\glsnavigation: new	150	instead of \csname	172
\glsxtr@org@getgrouptitle: new	149	\@glsxtr@dorecordnodefer: new	10
\glsxtr@recordsee: new	6	\@glsxtr@record@only@setup: added	
\glsxtr@writefields: added check for		check for \@gls@setupsort@none	15
automake	161	\@print@unsrt@glossary: corrected	
\glsxtrdisplayendloc: added check		misspelt command	166
for empty format	152	\@printunsrt@glossary@handler:	
\glsxtrgetgrouptitle: new	149	new	170
\glsxtrinitwrgloss: new	80	\gls@checkseeallowed: added	
\glsxtrlocationhyperlink: new	153	redefinition	31
\glsxtrsetgrouptitle: new	150	\glsxtr@writefields: added	
\glsxtrsupphypernumber: new	153	\providecommand lines	160
\ifglsxtrwrglossbefore: new	80	\glsxtrautoindex: new	217
1.15 (2017-05-10)		\glsxtrautoindexassignsort: new	217
\@glsxtr@dorecord: corrected		\glsxtrautoindexentry: new	217
premature expansion of \@glslocref	9	\glsxtrindexseealso: new	62
short-em-long-em: fixed spelling of		\glsxtrseealsolabels: new	65
\glsabbrvfont	312	\glsxtrseelist: new	61
short-long: fixed spelling of		\glsxtruseseealso: new	60
\glsabbrvfont	256	\glsxtruseseealsoformat: new	61
short-long-user: fixed spelling of		\seealso: new	62
\glsabbrvfont	342	autoindex: new	17
short-postfootnote-desc: fixed		1.17 (2017-08-09)	
spelling of \glsabbrvfont	263	\@glsxtr@mark@wordseps: new	230
short-postlong-user: fixed spelling of		\@glsxtr@markwordseps: new	230
\glsabbrvfont	339	\@glsxtr@noidx@displaynumberlist:	
short-postlong-user-desc: fixed		replace hard-coded ?? with	
spelling of \glsabbrvfont	340	\glsxtrundeftag	148

\@glxtr@noidx@entrynumberlist: replace hard-coded ?? with \glxtrundeftag	149	\Glsxtrsubsequentfmt: new	249
\@glxtr@noidx@numberlistloop: replace hard-coded ?? with \glxtrundeftag	148	\glxtrsubsequentfmt: new	249
\@glxtrifhyphenstart: new	344	\Glsxtrsubsequentplfmt: new	249
General: removed some inconsistencies in the abbreviation styles	253	\glxtrword: new	230
\glabbrvhyphenfont: new	344	\glxtrwordsep: new	230
\glabbrvonlyfont: new	359	short-hyphen-long-hyphen: new ...	353
\glabbrvscfont: new	272	short-hyphen-long-hyphen-desc: new	354
\glabbrvsmfont: new	289	short-hyphen-postlong-hyphen: new	356
\glabbrvuserfont: initialised to default font	331	short-hyphen-postlong-hyphen-desc: new	358
\glfirstabbrvhyphenfont: new ...	345	short-long-user-desc: corrected first forms	343
\glfirstabbrvonlyfont: new	359	short-nolong-desc-noreg: new ...	268
\glfirstabbrvscfont: new	273	short-nolong-noreg: new	266
\glfirstabbrvsmfont: new	289	long-em-noshort-em-desc-noreg: new	325
\glfirstlonghyphenfont: new ...	345	long-em-noshort-em-noreg: new ...	321
\glfirstlongonlyfont: new	359	long-hyphen-noshort-desc-noreg: new	347
\glslonghyphenfont: new	345	long-hyphen-noshort-noreg: new ..	349
\glslongonlyfont: new	359	long-hyphen-postshort-hyphen: new	350
\glslonguserfont: initialised to default font	331	long-hyphen-postshort-hyphen-desc: new	352
\glxtr@newabbreviation: added \glxtrorgshort and \glxtrorglong	231	long-hyphen-short-hyphen: new ...	345
\GlsXtrDefineAcShortcuts: new ...	21	long-hyphen-short-hyphen-desc: new	346
\glxtrgenabbrvfmt: added check for \ifglxtrininsertinside	247	long-noshort-desc-noreg: new ...	271
\glxtrhyphensuffix: new	345	long-noshort-noreg: new	272
\glxtrifhyphenstart: new	344	long-only-short-only: new	359
\glxtrlonghyphen: new	349	long-only-short-only-desc: new ..	361
\glxtrlonghyphennoshort: new ...	347	long-short-user-desc: corrected first forms	341
\glxtrlonghyphenshort: new	344	1.18 (2017-08-10) stylemods: changed default value to “default”	24
\glxtrlongshortdescname: new ...	254	1.19 (2017-09-09) \@glxtr@defaultnumberformat: new .	6
\glxtronlydescname: new	361	\@glxtr@dorecord: Use \@glxtr@recordloc instead of \@glxtr@recordloc	9
\glxtronlydescsort: new	361	\@glxtr@dorecordnodefer: Use \theglentrycounter for the location rather than \@glxtr@recordloc ..	10
\glxtronlysuffix: new	359	\@glxtr@record@setting: new	13
\glxtrtparen: new	234	\@glxtr@record@setting@alsoindex: new	13
\glxtrposthyphenlong: new	355	\@glxtrifhasfield: new	41
\glxtrposthyphenshort: new	350		
\glxtrposthyphensequent: new	350		
\glxtrshortdescname: new	266		
\glxtrshorthyphen: new	355		
\glxtrshorthyphenlong: new	353		
\glxtrshortlongdescname: new ...	257		
\glxtrshortlongdescsort: new ...	257		

General: added \glslink option		
theHvalue	80	
added \glslink option thevalue ...	80	
\glsxtr@writefields: removed		
double-quotes around \jobname ..	161	
\glsxtrdoautoindexname: changed		
format test	216	
\glsxtrhyperlink: new	109	
\glsxtrifhasfield: new	41	
\GlsXtrSetDefaultNumberFormat:		
new	6	
\s@glsxtrifhasfield: new	41	
1.20 (2017-09-11)		
\@glsxtrhypernameprefix: new	145	
\glsdohypertarget: added redefinition	146	
\printunrtglossaryunitsetup:		
switched from redefining		
\glolinkprefix to		
\@glsxtrhypernameprefix	171	
1.21 (2017-11-03)		
\@glsxtr@record: added check for		
default options	8	
\@glsxtrwrglossmark: new	26	
\@glslink: changed \let to \def	109	
\@glsxtr@checkgroup: new	172	
\@glsxtr@defpostpunc: new	18	
\@glsxtr@do@record@wrglossary:		
new	7	
\@glsxtr@dosee@alsoindex@glossary:		
new	30	
\@glsxtr@doseeglossary: new	30	
\@glsxtr@noidx@do: removed code		
dealing with the group	173	
\@glsxtr@record@setting@off: new ..	14	
\@glsxtr@record@setting@only: new ..	13	
\@glsxtr@rglstrigger@record: new ..	177	
\@glsxtrglossentry: new	163	
\@glsxtrnewgls: new	174	
\@glsxtrsetaliasnoindex: changed to		
use \glsxtrifhasfield instead of		
\ifglshasfield	103	
\@glsxtrwrglossmark: new	26	
\@rGLS: new	180	
\@rGLS@: new	180	
\@rGLSpl: new	180	
\@rGLSpl@: new	180	
\@rGls: new	179	
\@rGls@: new	179	
\@rGlspl: new	180	
\@rGlspl@: new	180	
\@rgls: new	178	
\@rgls@: new	178	
\@rglspl: new	179	
\@rglspl@: new	179	
General: adjusted mcolalmtree	524	
modified index to remove hard coded		
\space	501	
modified list to remove hard coded		
\space	489	
moved conditional outside of		
\glsgroupskip	493–500	
new	527	
redefined altlistgroup to discourage		
breaks after group headings	491	
redefined altlisthypergroup to		
discourage breaks after group		
headings	492	
redefined almtreegroup to discourage		
breaks after group headings	519	
redefined almtreehypergroup to		
discourage breaks after group		
headings	519	
redefined indexgroup to discourage		
breaks after group headings	502	
redefined indexhypergroup to		
discourage breaks after group		
headings	502	
redefined listgroup to discourage		
breaks after group headings	491	
redefined listhypergroup to		
discourage breaks after group		
headings	491	
redefined mcolalmtreegroup to		
discourage breaks after group		
headings	524	
redefined mcolalmtreehypergroup to		
discourage breaks after group		
headings	525	
redefined mcolalmtreespannav to		
discourage breaks after group		
headings	525	
redefined mcolindexgroup to		
discourage breaks after group		
headings	520	
redefined mcolindexhypergroup to		
discourage breaks after group		
headings	521	

redefined mcolindexspannav to	\glstreechildprelocation: new ...	501
discourage breaks after group	\glstreeprelocation: new	501
headings	\glstriggerrecordformat: new	178
redefined mcoltreegroup to	\glseuseabbrvfont: new	247
discourage breaks after group	\glseuselongfont: new	247
headings	\glsxtr@do@alsoindex@wrglossary:	
redefined mcoltreehypergroup to	new	7
discourage breaks after group	\glsxtr@org@@do@wrglossary: new ..	31
headings	\glsxtr@org@dohyperlink: new	106
redefined mcoltreenonamegroup to	\glsxtr@setbookindexmark: new ...	532
discourage breaks after group	\glsxtrbookindexatendgroup: new ..	528
headings	\glsxtrbookindexbetween: new	528
redefined	\glsxtrbookindexbookmark: new ...	528
mcoltreenonamehypergroup to	\glsxtrbookindexcols: new	527
discourage breaks after group	\glsxtrbookindexcolspread: new ..	529
headings	\glsxtrbookindexfirstmark: new ..	533
redefined mcoltreenonamespannav to	\glsxtrbookindexfirstmarkfmt: new	533
discourage breaks after group	\glsxtrbookindexformatheader: new	528
headings	\glsxtrbookindexgroupskip: new ..	528
redefined mcoltreespannav to	\glsxtrbookindexlastmark: new ...	533
discourage breaks after group	\glsxtrbookindexlastmarkfmt: new	533
headings	\glsxtrbookindexmarkentry: new ..	532
redefined treegroup to discourage	\glsxtrbookindexname: new	527
breaks after group headings	\glsxtrbookindexparentchildsep:	
redefined treehypergroup to	new	528
discourage breaks after group	\glsxtrbookindexparentschildsep:	
headings	new	528
redefined treenonamegroup to	\glsxtrbookindexprelocation: new	527
discourage breaks after group	\glsxtrbookindexsubatendgroup:	
headings	new	528
redefined treenonamehypergroup to	\glsxtrbookindexsubbetween: new ..	528
discourage breaks after group	\glsxtrbookindexsubname: new	527
headings	\glsxtrbookindexsubprelocation:	
debug: new	new	527
\gglsetwidest: new	\glsxtrbookindexsubsubatendgroup:	
\gl:disablehyper: added check for	new	528
existence	\glsxtrbookindexsubsubbetween:	
changed to use \def rather than \let	new	528
\gl:enablehyper: changed to use \def	\glsxtrbookindexthepage: new	532
rather than \let	\glsxtrdetoklocation: new	176
\Glsfmtname: new	\glxtrenablerecordcount: new ...	176
\Glsfmtname: new	\glxtrglossentry: new	163
\glshex: new	\glxtrgroupfield: new	171
\glslstchildpostlocation: new ..	\Glsxtrheadname: new	370
\glslstchildprelocation: new ...	\glxtrheadname: new	370
\glslstprelocation: new	\GlsXtrIfFieldEqStr: new	46
\gl:navhyperlink: patched	\glxtriflabelinlist: new	170
\gl:seeitemformat: new	\glxtrifrecordtrigger: new	177
\gl:showtarget: new		

\glxtrindexseealso: added check that the entry exists	63	\rGLSpl: new	180
\glxtrinithyperoutside: new	81	\rGlspl: new	179
\GlsXtrLocationRecordCount: new ..	176	\rglspl: new	179
\glxtrnewgls: new	173, 175	\rGLSplformat: new	181
\glxtrnewGLSlike: new	175	\rGlsplformat: new	181
\glxtrnewglslike: new	175	\rglsplformat: new	181
\glxtrnewrgls: new	175	\s@glxtrifhasfield: switched from \ifdef to \ifundef	41
\glxtrnewrGLSlike: new	176	1.22 (2017-11-08)	
\glxtrnewrglslike: new	176	\@glxtr@nopostpunc: new	144
\glxtrprelocation: new	488, 527	\@glxtr@orgprintglossary: changed explicit \let for \nopostdesc to \glxtractivatenopost	143
\GlsXtrRecordCount: new	176	\@glxtrglossentryother: new	164
\glxtrrecordtriggervalue: new ..	177	\glossentrynameother: new	214
\glxtrresourcefile: now disables record key	159	\glseeitemformat: switched check from regular to short	58
\glxtrresourceinit: new	159	\glxtr@setaccessdisplay: new ...	213
\GlsXtrSetRecordCountAttribute: new	177	\glxtr@writefields: provide \glxtr@record in aux file	161
\Glsxtrtitlename: new	370	\glxtractivatenopost: new	144
\glxtrtitlename: new	370	\glxtrbookindexprelocation: removed check for no post dot	527
\glxtrtitleorpdforheading: new ..	366	\glxtrglossentryother: new	164
\GlsXtrTotalRecordCount: new	176	\glxtrnopostpunc: new	144
\glxtrwrglossmark: new	26	1.23 (2017-11-12)	
short-em: new	315	\@@glxtrfmt: added check for indexing added grouping	36
short-sc: corrected first letter uppercasing	277	new	36
short-sm: corrected first letter uppercasing	294	\@glxtr@nopostpunc@postdesc: new	144
shortcuts: ac	23	\@glxtr@restore@postpunc: new ..	145
\ifglxtr@hyperoutside: new	80	\@glxtrentryfmt: fixed missing label argument	37
all: new	487	\@glxtrfmt: new	35
nolong-short: new	268	\eglsupdatewidest: new	509
nolong-short-em: new	317	\gglupdatewidest: new	509
nolong-short-noreg: new	269	\glsupdatewidest: new	509
nolong-short-sc: new	280	\GlsXtrDefineAbbreviationShortcuts: changed \newabbr definition to use \providecommand	21
nolong-short-sm: new	296	\GlsXtrDefineAcShortcuts: changed \newabbr definition to use \providecommand	21
nopostdot: new	19	\glxtrfmtdisplay: new	36
postpunc: new	19	\glxtrifcustomdiscardperiod: new	225
\printunsrtglossaryentryprocesshook: new	170	\GlsXtrIfFieldUndef: new	43
\printunsrtglossarypredoglossary: new	170	\glxtrrestorepostpunc: new	145
\printunsrtglossaryskipentry: new	170	\s@glxtrfmt: new	36
\rGLS: new	180	\s@glxtrfmt: new	36
\rGls: new	179		
\rgls: new	178		
\rGLSformat: new	181		
\rGlsformat: new	181		
\rglsformat: new	181		

\xglupdatewidest:new	510	\glxtrcombiningdiacriticIrules:	
1.24 (2017-11-14)		new	457
\glsadd:added \@gls@setsort	85	\glxtrcombiningdiacriticIVrules:	
\glxtrforcsvfield:new	38	new	459
\glxtrlocalsetgrouptitle:new ..	150	\glxtrcombiningdiacriticrules:	
1.25 (2017-11-14)		new	457
\glxtrbookindexmulticolseenv:new	529	\glxtrcontrolrules:new	456
1.25 (2017-11-24)		\glxtrcurrencyrules:new	461
\glsextrapostnamehook:new	213	\glxtrdigitrules:new	484
\glxtrfootnotename:new	258	\glxtrfractionrules:new	485
\glxtrlongnoshortdescname:new ..	269	\glxtrGeneralLatinIIIrules:new	463
\glxtrlongnoshortname:new	271	\glxtrGeneralLatinIIrules:new ..	462
\glxtrlongshortname:new	253	\glxtrGeneralLatinIrules:new ..	462
\glxtrlongshortuserdescname:new	336	\glxtrGeneralLatinIVrules:new ..	464
\glxtronlyname:new	359	\glxtrGeneralLatinVIIrules:new	467
\glxtrpostlinkAddDescOnFirstUse:		\glxtrGeneralLatinVIrules:new ..	465
changed to use \glxtrparen	226	\glxtrGeneralLatinVrules:new ..	464
\glxtrpostlinkAddSymbolOnFirstUse:		\glxtrgeneralpuncIIrules:new ..	461
changed to use \glxtrparen	226	\glxtrgeneralpuncIrules:new ...	460
\glxtrshortlongname:new	255	\glxtrgeneralpuncrules:new	460
\glxtrshortlonguserdescname:new	340	\glxtrhyphenrules:new	460
\glxtrshortnolongname:new	264	\glxtrLatinA:new	467
1.26 (2018-01-05)		\glxtrLatinAA:new	469
\@glxtr@do@inc@linkcount:new ..	182	\glxtrLatinAELigature:new	469
\glslinkpresetkeys:new	81	\glxtrLatinE:new	467
\glxtr@inc@linkcount:new	81	\glxtrLatinEszettSs:new	469
\GlsXtrEnableLinkCounting:new ..	183	\glxtrLatinEszettSz:new	469
\GlsXtrIfLinkCounterDef:new	183	\glxtrLatinEth:new	469
\glxtrinclinkcounter:new	182	\glxtrLatinH:new	467
\GlsXtrLinkCounterName:new	183	\glxtrLatinI:new	468
\GlsXtrLinkCounterValue:new	182	\glxtrLatinInsularG:new	470
\GlsXtrTheLinkCounter:new	182	\glxtrLatinK:new	468
1.27 (2018-02-26)		\glxtrLatinL:new	468
\@glxtrdialecthook:new	33	\glxtrLatinLslash:new	470
General: added		\glxtrLatinM:new	468
glossaries-extra-bib2gls.sty	437	\glxtrLatinN:new	468
\Alpha:new	454	\glxtrLatinO:new	468
\Beta:new	454	\glxtrLatinOELigature:new	469
\Chi:new	454	\glxtrLatinOslash:new	470
\Digamma:new	455	\glxtrLatinP:new	468
\Epsilon:new	454	\glxtrLatinS:new	468
\Eta:new	454	\glxtrLatinSchwa:new	469
\glxtr@loaddialect:new	437	\glxtrLatinT:new	468
\glxtrBasicDigitrules:new	484	\glxtrLatinThorn:new	469
\glxtrcombiningdiacriticIIrules:		\glxtrLatinWynn:new	470
new	458	\glxtrLatinX:new	469
\glxtrcombiningdiacriticIrules:		\glxtrMathGreekIIrules:new	476
new	458	\glxtrMathGreekIrules:new	475

<code>\glxtrMathItalicAlpha</code> : new	480	<code>\glxtrUpEpsilon</code> : new	478
<code>\glxtrMathItalicBeta</code> : new	480	<code>\glxtrUpEta</code> : new	478
<code>\glxtrMathItalicChi</code> : new	483	<code>\glxtrUpGamma</code> : new	477
<code>\glxtrMathItalicDelta</code> : new	481	<code>\glxtrUpIota</code> : new	478
<code>\glxtrMathItalicEpsilon</code> : new	481	<code>\glxtrUpKappa</code> : new	478
<code>\glxtrMathItalicEta</code> : new	481	<code>\glxtrUpLambda</code> : new	479
<code>\glxtrMathItalicGamma</code> : new	481	<code>\glxtrUpMu</code> : new	479
<code>\glxtrMathItalicGreekIIrules</code> :		<code>\glxtrUpNu</code> : new	479
new	472	<code>\glxtrUpOmega</code> : new	480
<code>\glxtrMathItalicGreekIrules</code> : new	471	<code>\glxtrUpOmicron</code> : new	479
<code>\glxtrMathItalicIota</code> : new	481	<code>\glxtrUpPhi</code> : new	480
<code>\glxtrMathItalicKappa</code> : new	482	<code>\glxtrUpPi</code> : new	479
<code>\glxtrMathItalicLambda</code> : new	482	<code>\glxtrUpPsi</code> : new	480
<code>\glxtrMathItalicLowerGreekIIrules</code> :		<code>\glxtrUpRho</code> : new	479
new	474	<code>\glxtrUpSigma</code> : new	479
<code>\glxtrMathItalicLowerGreekIrules</code> :		<code>\glxtrUpTau</code> : new	480
new	473	<code>\glxtrUpTheta</code> : new	478
<code>\glxtrMathItalicMu</code> : new	482	<code>\glxtrUpUpsilon</code> : new	480
<code>\glxtrMathItalicNabla</code> : new	484	<code>\glxtrUpXi</code> : new	479
<code>\glxtrMathItalicNu</code> : new	482	<code>\glxtrUpZeta</code> : new	478
<code>\glxtrMathItalicOmega</code> : new	483	<code>\Iota</code> : new	454
<code>\glxtrMathItalicOmicron</code> : new	482	<code>\Kappa</code> : new	454
<code>\glxtrMathItalicPartial</code> : new	484	<code>\Mu</code> : new	454
<code>\glxtrMathItalicPhi</code> : new	483	<code>\Nu</code> : new	454
<code>\glxtrMathItalicPi</code> : new	482	<code>\Omicron</code> : new	454
<code>\glxtrMathItalicPsi</code> : new	483	<code>\omicron</code> : new	455
<code>\glxtrMathItalicRho</code> : new	482	<code>\Rho</code> : new	454
<code>\glxtrMathItalicSigma</code> : new	483	<code>\Tau</code> : new	454
<code>\glxtrMathItalicTau</code> : new	483	<code>\Upalpha</code> : new	455
<code>\glxtrMathItalicTheta</code> : new	481	<code>\Upbeta</code> : new	455
<code>\glxtrMathItalicUpperGreekIIrules</code> :		<code>\Upchi</code> : new	456
new	473	<code>\Upepsilon</code> : new	455
<code>\glxtrMathItalicUpperGreekIrules</code> :		<code>\Upeta</code> : new	455
new	472	<code>\Upiota</code> : new	455
<code>\glxtrMathItalicUpsilon</code> : new	483	<code>\Upkappa</code> : new	455
<code>\glxtrMathItalicXi</code> : new	482	<code>\Upmu</code> : new	455
<code>\glxtrMathItalicZeta</code> : new	481	<code>\Upnu</code> : new	455
<code>\glxtrMathUpGreekIIrules</code> : new	471	<code>\Upomicron</code> : new	455
<code>\glxtrMathUpGreekIrules</code> : new	470	<code>\upomicron</code> : new	456
<code>\glxtrnonprintablerules</code> : new	457	<code>\Uprho</code> : new	455
<code>\glxtrprovidecommand</code> : new	439	<code>\Uptau</code> : new	455
<code>\glxtrspacerules</code> : new	457	<code>\Upzeta</code> : new	455
<code>\glxtrSubScriptDigitrules</code> : new	484	<code>\Zeta</code> : new	454
<code>\glxtrSuperScriptDigitrules</code> : new	484		
<code>\glxtrUpAlpha</code> : new	477		
<code>\glxtrUpBeta</code> : new	477		
<code>\glxtrUpChi</code> : new	480		
<code>\glxtrUpDelta</code> : new	477		
<code>\glxtrUpDigamma</code> : new	478		

1.28 (2018-03-06)	
<code>\@glxtr@docdefval</code> : changed from	
count register to macro	16
<code>\@glxtrdialecthook</code> : save and restore	
<code>\TrackLangRequireDialectPrefix</code>	
	485

\glstredfield: changed \csedef to \protected\csedef	44	\glsaddpresetkeys: new	85
\glsxtrlocalsetgrouptitle: changed \csedef \protected\csedef	150	\glsuserdescription: new	331
\glsxtrsetgrouptitle: changed \csxdef \protected\csxdef	150	\glsxtrabbreviationfont: new	77
1.29 (2018-04-09)		\GlsXtrDualBackLink: new	440
\@gls@removespaces: added expansion	153	\GlsXtrDualField: new	440
\@glsxtr@dorecord: don't suppress expansion of \@glsrecordlocref if counter isn't page	10	\GlsXtrExpandedFmt: new	81
\@glsxtr@wrglossary@locationhyperlink: new	25	\GLSxtrlong: added \@glsxtr@record	242
\glsxtr@inc@wrglossaryctr: new ...	25	\Glsxtrlong: added \@glsxtr@record	242
\glsxtr@wrglossarylocation: new ..	440	\glsxtrlong: added \@glsxtr@record	241
\GlsXtrBibTeXEntryAliases: new ..	441	\GLSxtrlongpl: added \@glsxtr@record	246
\glsxtrfieldforlistloop: corrected argument order in \forlistcsloop	38	\Glsxtrlongpl: added \@glsxtr@record	245
\GlsXtrIndexCounterLink: new	440	\glsxtrlongpl: added \@glsxtr@record	245
\GlsXtrInternalLocationHyperlink: new	25	\GLSxtrshort: added \@glsxtr@record	240
\GlsXtrProvideBibTeXFields: new ..	441	\Glsxtrshort: added \@glsxtr@record	240
indexcounter: new	26	\glsxtrshort: added \@glsxtr@record	239
\setentrycounter: new	152	\GLSxtrshortpl: added \@glsxtr@record	244
1.30 (2018-04-25)		\Glsxtrshortpl: added \@glsxtr@record	244
\@glsxtr@record: added check for post-key hook	8	\glsxtrshortpl: added \@glsxtr@record	243
added check for pre-key hook	8	\GlsXtrStartUnsetBuffering: new ..	119
\@GLSxtr@fullpl: added \@glsxtr@record	239	\GlsXtrStopUnsetBuffering: new ..	119
\@GlsXtrStopUnsetBuffering: new ..	120	indexcounter: added check for wrglossary counter	26
\@Glsxtr@fullpl: added \@glsxtr@record	238	\s@GlsXtrStopUnsetBuffering: new	120
\@glsxtr@dorecord: don't suppress expansion of \@glsrecordlocref ..	10	1.31 (2018-05-09)	
\@glsxtr@full: added \@glsxtr@record	236	\@GlsXtrStartUnsetBuffering: new	119
\@glsxtr@fullpl: added \@glsxtr@record	238	\@gls@ifaccessattribute@set: new	192
\@glsxtr@glossadd@postkeys: new ...	9	\@gls@initaccesskeys: new ...	191, 200
\@glsxtr@glossadd@prekeys: new	9	\@gls@setup@default@short@access: new	193
\@glsxtr@glslink@postkeys: new	9	\@glsxtr@record@noglossarywarning: new	158
\@glsxtr@glslink@prekeys: new	9	\@glsxtrbuffer@nodup@unset: new ..	119
\@glsxtr@local@textformat: new ...	80	General: added prefix key for glslink	81
\@glsxtr@unset: new	118	added prefix key for printgloss ..	145
\@glsxtrbuffer@unset: new	119	changed \let to \def	145
\glsadd: added \glsaddpostsetkeys ..	85	\glsaddeach: new	86
added \glsaddpresetkeys	85	\glsapturedgroup: new	439
\glsaddpostsetkeys: new	85	\glsdefpostdesc: new	224
		\glsdefpostlink: new	225
		\glsdefpostname: new	213

\glsdohypertarget: bug fix: ensure that new version is picked up	146	\if@glxtrdocdefrestricted: changed to allow for atom as well ...	17
\glslistdesc: new	489	docdef: atom	17
\glslocalreseteach: new	121	1.35 (2018-08-13)	
\glslocalunseteach: new	121	\@glxtr@link@: initialise post-link hook commands	79
\glstreechilddesc: new	504	1.36 (2018-08-18)	
\glstreechildsymbol: new	504	\glxtrautoindexesc: new	217
\glstreedefaultnamefmt: new	500	\glxtrdisplaysupplc: new	442
\glstreedesc: new	503	\glxtrmultisupplcation: new ...	441
\glstreegroupheaderfmt: added redefinition	501	1.37 (2018-11-30)	
\glstreenamefmt: added redefinition .	501	\@glxtr@record: added check for auto-add	8
\glstreenavigationfmt: added redefinition	501	\@dGLS: new	450
\glstreenonamechilddesc: new	506	\@dGLSpl: new	450
\glstreenonamedesc: new	505	\@dGls: new	450
\glstreenonamesymbol: new	506	\@dGlspl: new	450
\glstreesymbol: new	504	\@dgls: new	449
\glxtr@newabbreviation: added \ExtraCustomAbbreviationFields	231	\@dglspl: new	449
\GlsXtrForUnsetBufferedList: new	120	\@glxtr@getcounterprefix: new	32
\GlsXtrIfFieldCmpNum: new	42	\@glxlongextrawidestname: new ...	536
\GlsXtrIfFieldEqNum: new	42	\@glxtr@bibgls@removespaces: new	444
\GlsXtrIfFieldEqXpStr: new	46	\@glxtr@check@bibgls@nameref: new	159
\GlsXtrIfFieldNonZero: new	42	\@glxtr@do@nameref@record: new ..	11
\GlsXtrIfHasNonZeroChildCount: new	439	\@glxtr@get@prefixedlabel: new .	449
\GlsXtrIfXpFieldEqXpStr: new	47	\@glxtr@if@record@only: new	14
\glxtrpostlinkAddSymbolDescOnFirstUse: new	226	\@glxtr@ifnum@mmode: new	11
\GlsXtrRecordWarning: new	157	\@glxtr@labelprefixes: new	448
\glxtrRevertToCmarks: new	365	\@glxtr@prefixlabellist: new ...	448
\GlsXtrStandaloneGlossaryType: new	163	\@glxtr@providenewgls: new	174
\GlsXtrStandaloneSubEntryItem: new	164	\@glxtr@record@only@setup: new ..	14
\s@GlsXtrStartUnsetBuffering: new	119	\@glxtr@record@setting@nameref: new	14
1.32 (2018-05-24)		\@glxtr@use@equation@counter@or: new	81
\GlsXtrForeignText: new	47	General: new	534
\GlsXtrForeignTextField: new	49	page: nameref	10
\GlsXtrUnknownDialectWarning: new	49	\dGLS: new	450
1.33 (2018-07-26)		\dgls: new	449
\ifglsused: added redefinition	51	\dglldisp: new	450
1.34 (2018-07-29)		\dglslink: new	450
\gls@begindocdefs: atom	67	\dGLSpl: new	450
\GlsXtrIfUnusedOrUndefined: new ..	33	\glsadd: added grouping	85
\glxtrNoGlossaryWarning: added package warning	24	ensure that \glsadd performs indexing	86
		\glslongextraDescAlign: new	536
		\glslongextraDescFmt: new	534
		\glslongextraDescNameHeader: new	541

\glslongextraDescNameTabularFooter:	\glslongextraNameDescSymTabularFooter:
new 541	new 544
\glslongextraDescNameTabularHeader:	\glslongextraNameDescSymTabularHeader:
new 541	new 544
\glslongextraDescSymNameHeader:	\glslongextraNameDescTabularFooter:
new 553	new 536
\glslongextraDescSymNameTabularFooter:	\glslongextraNameDescTabularHeader:
new 553	new 536
\glslongextraDescSymNameTabularHeader:	\glslongextraNameFmt: new 534
new 553	\glslongextraNameSymDescHeader:
\glslongextraGroupHeading: new .. 536	new 547
\glslongextraHeaderFormat: new .. 536	\glslongextraNameSymDescLocationHeader:
\glslongextraLocationAlign: new . 536	new 548
\glslongextraLocationDescNameHeader:	\glslongextraNameSymDescLocationTabularFooter:
new 542	new 549
\glslongextraLocationDescNameTabularFooter:	\glslongextraNameSymDescLocationTabularHeader:
new 542	new 548
\glslongextraLocationDescNameTabularHeader:	\glslongextraNameSymDescTabularFooter:
new 542	new 547
\glslongextraLocationDescSymNameHeader:	\glslongextraNameSymDescTabularHeader:
new 554	new 547
\glslongextraLocationDescSymNameTabularFooter:	\glslongextraSetDescWidth: new .. 537
new 555	\glslongextraSetWidest: new 536
\glslongextraLocationDescSymNameTabularHeader:	\glslongextraSubDescFmt: new 535
new 555	\glslongextraSubLocationFmt: new 535
\glslongextraLocationFmt: new ... 535	\glslongextraSubNameFmt: new 535
\glslongextraLocationSymDescNameHeader:	\glslongextraSubSymbolFmt: new .. 535
new 551	\glslongextraSymbolAlign: new ... 536
\glslongextraLocationSymDescNameTabularFooter:	\glslongextraSymbolFmt: new 534
new 552	\glslongextraSymDescNameHeader:
\glslongextraLocationSymDescNameTabularHeader:	new 550
new 551	\glslongextraSymDescNameTabularFooter:
\glslongextraLocSetDescWidth: new 538	new 550
\glslongextraNameAlign: new 536	\glslongextraSymDescNameTabularHeader:
\glslongextraNameDescHeader: new 536	new 550
\glslongextraNameDescLocationHeader:	\glslongextraSymLocSetDescWidth:
new 539	new 538
\glslongextraNameDescLocationTabularFooter:	\glslongextraSymSetDescWidth: new 537
new 540	\glslongextraTabularVAlign: new . 538
\glslongextraNameDescLocationTabularHeader:	\glslongextraUpdateWidest: new .. 537
new 539	\glslongextraUpdateWidestChild:
\glslongextraNameDescSymHeader:	new 537
new 544	\glsrenewcommand: new 439
\glslongextraNameDescSymLocationHeader:	\glsseeitemformat: removed reference
new 545	to \glslabel 58
\glslongextraNameDescSymLocationTabularFooter:	\glsxtr@dblfloat: new 18
new 545	\glsxtr@do@autoadd: new 81
\glslongextraNameDescSymLocationTabularHeader:	\glsxtr@float: new 18
new 545	\glsxtr@record@nameref: new 162

\glstr@renewcommand: new	439	1.38 (2018-12-01)	
\glstr@writefields: provide			\glslongextraNameFmt: bug fix:
\glstr@record@nameref in aux			removed double param
file	161		all: added glossary-longextra
\glstraddlabelprefix: new	448	1.39 (2019-03-22)	
\GlsXtrAutoAddOnFormat: new	81		\@GlsXtrIfFieldCmpNum: new
\glstrclearlabelprefixes: new	448		\@GlsXtrIfFieldEqNum: new
\glstrdisplaylocnameref: new	442		\@GlsXtrIfFieldEqStr: new
\glstrfmtexternalnameref: new	444		\@GlsXtrIfFieldEqXpStr: new
\glstrfmtinternalnameref: new	444		\@GlsXtrIfFieldNonZero: new
\GLSXRhiername: new	60		\@GlsXtrIfXpFieldEqXpStr: new
\GLSxtrhiername: new	59		\@glstr@removespaces: changed \x to
\GlsXtrhiername: new	59		\@glo@tmp
\Glsxtrhiername: new	59		\@glstr@dorecord: added protection
\glstrhiername: new	58		for fragile commands
\glstrhiernamesep: new	60		General: added label key for
\glstridentifyglsl-like: new	174		printgloss
\glstrifinlabelprefixlist: new	448		\glstrbookindexlocation: new
\GlsXtrLocationField: new	172		\glstrbookindexsublocation: new
\glstrnameloclink: new	443		\glstrentryparentname: new
\glstrnamerefink: new	443		\GlsXtrIfFieldCmpNum: added starred
\glstrprependlabelprefix: new	448		version
\GlsXtrSetAltModifier: write modifier			\GlsXtrIfFieldEqNum: added starred
to aux	106		version
\glstrSetWidest: new	445		\GlsXtrIfFieldEqStr: added starred
\glstrSetWidestFallback: new	447		form
\GlsXtrStandaloneEntryName: new	163		\GlsXtrIfFieldEqXpStr: added starred
\GlsXtrStandaloneEntryOther: new	165		form
\GLSxtrusefield: new	44		\GlsXtrIfFieldNonZero: added starred
\Glsxtrusefield: fixed internal			version
command and added check for			\GlsXtrIfXpFieldEqXpStr: added
\texorpdfstring	43		starred form
\ifGlsLongExtraUseTabular: new	538		\glstrsetglossarylabel: new
floats: new	18		show long form as advertised in the
long-desc-name: new	541		manual
long-desc-sym-name: new	553		short-desc: corrected to omit
long-loc-desc-name: new	543		description key as advertised in the
long-loc-desc-sym-name: new	555		manual
long-loc-sym-desc-name: new	552		short-em-desc: bug fix: omit description
long-name-desc: new	538		key as advertised in the manual
long-name-desc-loc: new	540		short-sc-desc: bug fix: omit description
long-name-desc-sym: new	544		key as advertised in the manual
long-name-desc-sym-loc: new	546		short-sm-desc: corrected to omit
long-name-sym-desc: new	547		description key as advertised in the
long-name-sym-desc-loc: new	549		manual
long-sym-desc-name: new	550		\s@GlsXtrIfFieldCmpNum: new
equations: new	18		\s@GlsXtrIfFieldEqNum: new
			\s@GlsXtrIfFieldEqStr: new

\s@GlsXtrIfFieldEqXpStr: new	46	1.41 (2019-04-09)	General: changed \thisgrptitle to	
\s@GlsXtrIfXpFieldEqXpStr: new . . .	47		\glxtrcurrentgrptitle	532
1.4.2 (??)			\glslistgroupskip: new	489
\@glossentrysymbol: new	221		\glstopicAssignSubIndent: moved	
\glstentrypdfsymbol: new	221		\par from \glstopicSubItem	559
1.40 (2019-03-22)			\glstopicSubItem: added check for	
General: new	557		description	560
\glstopicAssignSubIndent: new . . .	559		moved \par to	
\glstopicAssignWidest: new	560		\glstopicAssignSubIndent	560
\glstopicCols: new	561		\glstopicSubLoc: moved \space to	
\glstopicColsEnv: new	561		\glstopicSubPreLocSep	561
\glstopicDesc: new	559		\glstopicSubPreLocSep: new	561
\glstopicGroupHeading: new	558		\glstreeChildDescLoc: new	504
\glstopicInit: new	559		\glstreeDescLoc: new	503
\glstopicItem: new	558		\glstreegroupskip: new	501
\glstopicLoc: new	559		\glstreePreHeader: new	501
\glstopicMarker: new	558		\glxtralttreeSymbolDescLocation:	
\glstopicMidSkip: new	560		added check for description	508
\glstopicName: new	558		topic: added penalty if no description .	557
\glstopicParIndent: new	559		topicmcols: added penalty if no	
\glstopicPostSkip: new	560		description	562
\glstopicPreSkip: new	560	1.42 (2020-02-03)		
\glstopicSubIndent: new	559		\@@glxtr@record: moved label	
\glstopicSubItem: new	560		definition outside of conditional	8
\glstopicSubItemBox: new	561		\@ACRfull: added redefinition	99
\glstopicSubItemSep: new	561		\@ACRfullpl: added redefinition	99
\glstopicSubLoc: new	561		\@Acrfull: added redefinition	99
\glstopicSubNameFont: new	561		\@Acrfullpl: added redefinition	99
\glstopicTitleFont: new	559		\@GlsXtrIfFieldValueInCsvList:	
\glstopicwidest: new	559		new	39
all: added glossary-topic	487		\@acrfull: added redefinition	98
topic: new	557		\@acrfullpl: added redefinition	99
topicmcols: new	561		\@domakeglossaries: provided	
1.40 (2019-03-31)			definition for \@domakeglossaries	138
\glsfirstabbrvdefaultfont: changed			\@gls@assign@actual: new	192
definition from \glsabbrvfont to			\@gls@entry@field: redefined	51
\glsabbrvdefaultfont for			\@gls@setup@default@access: added	
consistency	235		\glsdefaultshortaccess	193
\GlsXtrDefaultResourceOptions:			\@gls@setup@default@short@access:	
new	158		renamed to	
long-hyphen-noshort-noreg:			\@gls@setup@default@access	193
corrected formatting commands . . .	349		\@glslink: switched from	
\printunsrtabbreviations: new . . .	438		\glsdohyperlink to	
\printunsrtacronyms: new	438		\glxtrdohyperlink	109
\printunsrtindex: new	438		\@glxtr@abbrlists: new	135
\printunsrtindex: new	438		\@glxtr@acronymlists: new	135
\printunsrtnumbers: new	438		\@glxtr@addabbreviationlist: new	136
\printunsrtsymbols: new	438		\@glxtr@base@acrcmd: new	94

\@glxtr@doloadprefix: new	23	removed \relax and updated	
\@glxtr@org@addtoacronymlists:		\@gls@short instead of	
new	135	\glsshorttok	232
\@glxtr@org@setacronymlists: new	135	replaced explicit \spacefactor with	
\@glxtr@entryfmt: added \glslabel		\@	232
and scope	37	\glxtr@writefields: added check for	
General: added \@afterheading	521	order=letter	162
debug: showaccsupp	27	\glxtrAccSuppAbbrSetFirstLongAttrs:	
\forall abbreviations: new ...	135	new	196, 200
\forall acronyms: new	136	\glxtrAccSuppAbbrSetNameLongAttrs:	
\glsdefaultshortaccess: new	192	new	197, 200
\glsdisplaynumberlist: added	438	\glxtrAccSuppAbbrSetNameShortAttrs:	
\glsenablehyper: switched from		new	196, 200
\glsdohyperlink to		\glxtrAccSuppAbbrSetNoLongAttrs:	
\glxtrdohyperlink	109	new	196, 200
\glstrynumberlist: added	439	\glxtrAccSuppAbbrSetTextShortAttrs:	
\GLSfmtfirst: new	383	new	196, 200
\GLSfmtfirstpl: new	383	\glxtralttreeSymbolDescLocation:	
\GLSfmtfull: new	386	switched to using \glstreeDescLoc	508
\GLSfmtfull: switched pdf case to use		\glxtrassignactualsetup: new ...	192
\glspdffmtfull	386	\glxtrbookindexbookmarkprefix:	
\glspdffmtfull: switched pdf case to use		new	529
\glspdffmtfull	386	\GlsXtrDiscardUnsetBuffering: new	120
\GLSfmtfullpl: new	387	\glxtrdohyperlink: new (was former	
\GLSfmtfullpl: switched pdf case to use		redefinition of \glsdohyperlink)	108
\glspdffmtfullpl	387	\glxtrrequeationlocfmt: new	443
\glspdffmtfullpl: switched pdf case to use		\glxtrfieldformatcsvlist: new ...	39
\glspdffmtfullpl	386	\glxtrfieldformatlist: new	38
\GLSfmtlong: new	384	\glxtrfootnotedescname: new ...	260
\GLSfmtlongpl: new	385	\glxtrfootnotedescsort: new ...	260
\GLSfmtname: new	380	\GLSXtrhiername: switched to using	
\GLSfmtplural: new	382	\GLSfmtttext and \GLSfmtname	60
\GLSfmtttext: new	381	\GLSXtrhiername: switched to using	
\glspdffmtfull: new	385	\glsfmmttext, \glsfmtname,	
\glspdffmtfullpl: new	386	\GLSfmtttext and \GLSfmtname	59
\glseeitemformat: switched to using		\GlsXtrhiername: switched to using	
\glsfmmttext and \glsfmtname	58	\glsfmmttext and \glsfmtname	59
\glsshowtarget: added check for		\GlsXtrhiername: switched to using	
\glsshowtargetouter	29	\glsfmmttext and \glsfmtname	59
\glstreeChildDescLoc: added		\glxtrhiername: switched to using	
\glstreeNoDescSymbolPreLocation		\glsfmmttext and \glsfmtname	58
.....	504	\GlsXtrIfFieldValueInCsvList: new	39
\glstreegroupheaderskip: new ...	501	\glxtrpdfentryfmt: new	37
\glstreeNoDescSymbolPreLocation:		\glxtrprovideaccsuppcmd: new ...	196
new	503	\glxtrscsuffix: added \protect ..	273
\glxtr@newabbreviation: moved		\GlsXtrSetAltModifier: added check	106
apply abbreviation style to after		\GLSXtrtitlefirst: new	373
category key has been obtained	231	\GLSXtrtitlefirstplural: new ...	374
		\GLSXtrtitlefull: new	378

\GLSxtrtitlefullpl: new	378	moved \protect inside	
\GLSxtrtitlelong: new	376	\glxtrscsuffix	284
\GLSxtrtitlelongpl: new	376	short-sc-footnote-desc: new	285
\GLSxtrtitlename: new	371	short-sc-long: added missing text key	275
\GLSxtrtitleplural: new	372	moved \protect inside	
\GLSxtrtitleshort: new	369	\glxtrscsuffix	275
\GLSxtrtitleshortpl: new	370	short-sc-postfootnote: added	
\GLSxtrtitletext: new	371	missing text key	286
\glxtrusealias: new	61	moved \protect inside	
short-em: removed \protect from		\glxtrscsuffix	287
\glxtremsuffix	314	short-sc-postfootnote-desc: new .	288
short-em-desc: removed \protect		short-sm: removed \protect from	
from \glxtremsuffix	316	\glxtrsmsuffix	294
short-em-footnote: added missing text		short-sm-desc: removed \protect	
key	325	from \glxtrsmsuffix	295
removed \protect from		short-sm-footnote: added missing text	
\glxtremsuffix	326	key	300
short-em-footnote-desc: new	327	removed \protect from	
short-em-long: added missing text key	310	\glxtrsmsuffix	301
removed \protect from		short-sm-footnote-desc: new	302
\glxtremsuffix	311	short-sm-long: added missing text key	291
short-em-long-em: added missing text		removed \protect from	
key	312	\glxtrsmsuffix	292
removed \protect from		short-sm-postfootnote: added	
\glxtremsuffix	313	missing text key	303
short-em-postfootnote: added		removed \protect from	
missing text key	328	\glxtrsmsuffix	304
removed \protect from		short-sm-postfootnote-desc: new .	305
\glxtremsuffix	329	\makeglossaries: added	
short-em-postfootnote-desc: new .	330	\@domakeglossaries	139
short-footnote-desc: new	260	let \@makeglossary to \@gobble	
short-hyphen-long-hyphen: added		instead of \relax	140
missing text key	353	removed redefinition of	
short-hyphen-postlong-hyphen:		\makeglossary	140
added missing text key	356	\makenoidxglossaries: added	
short-long: added missing text key ...	256	\@domakeglossaries	68
short-long-user: added missing text		long-em-noshort-em: removed	
key	342	\protect from \glxtremsuffix .	320
short-postfootnote-desc: added		long-em-noshort-em-desc: removed	
missing text key	263	\protect from \glxtremsuffix .	323
new	263	long-em-short-em: added missing text	
short-postlong-user: added missing		key	308
text key	338	removed \protect from	
short-sc: moved \protect inside		\glxtremsuffix	309
\glxtrscsuffix	277	long-hyphen-noshort-desc-noreg:	
short-sc-desc: moved \protect inside		added missing text key	347
\glxtrscsuffix	279	long-hyphen-postshort-hyphen:	
short-sc-footnote: added missing text		added missing text key	350
key	284		

long-hyphen-short-hyphen: added missing text key	345	\@glxtr@noidx@do: replaced \ifglshasparent with \@glxtr@ifischild	172
long-noshort-em: removed \protect from \glxtremsuffix	318	\@print@unsrt@innerglossary: new	168
long-noshort-em-desc: removed \protect from \glxtremsuffix .	322	General: added groups key	146
long-noshort-sc: moved \protect inside \glxtrscsuffix	281	added leveloffset key	146
long-noshort-sc-desc: moved \protect inside \glxtrscsuffix	282	\doifglossarynoexistsordo: switched to starred form of \ifglossaryexists	57
long-noshort-sm: removed \protect from \glxtrsmsuffix	297	\glswriteentry: replaced \ifglused with \GlsXtrIfUnusedOrUndefined ..	104
long-noshort-sm-desc: removed \protect from \glxtrsmsuffix .	299	\glxtr@printgloss@checkexists: new	143
long-only-short-only: added missing text key	359	\glxtralttreeSymbolDescLocation: removed duplicate description	508
removed \protect from \glxtronlysuffix	360	\ifglossaryexists: added check for starred form	33
long-postshort-user: added missing text key	333	\np@glxtr@assign@leveloffset: new	146
long-short: added missing text key ...	253	\p@glxtr@assign@leveloffset: new	146
long-short-em: added missing text key	306	\pp@glxtr@assign@leveloffset: new	146
removed \protect from \glxtremsuffix	307	\printunsrtglossary: added check for \@printgloss@checkexists	165
long-short-sc: added missing text key	273	\printunsrtinnerglossary: new ...	167
moved \protect inside \glxtrscsuffix	273	printunsrtglossarywrap: new	167
long-short-sm: added missing text key	290	1.45 (2020-04-01)	
removed \protect from \glxtrsmsuffix	290	General: removed duplicate description	506
long-short-user: added missing text key	332	\glstreenonameChildDescLoc: new .	506
footnote: added missing text key	258	\glstreenonameDescLoc: new	506
footnote-desc: new	261	1.46 (2021-09-18)	
postfootnote: added missing text key .	261	\@glxtrsetaliasnoindex: changed to use starred version of \glxtrifhasfield	103
prefix: new	23	1.46 (2021-09-20)	
\RestoreAcronyms: added display style	137	\@@glxtr@record: changed \edef to \protected@edef	8
\s@GlsXtrIfFieldValueInCsvList: new	39	\@@newglossaryentry@defunitcounters: changed \edef to \protected@edef	127
\seealso: add check for \also	62	\@glossentrysymbol: changed \edef to \protected@edef	221
1.42 (?)		\@gl@increment@currunitcount: changed \edef to \protected@edef	128
postfootnote-desc: new	264	\@gl@link: changed \edef to \protected@edef	82, 83
1.43 (2020-02-28)		\@gl@link@checkfirsthyper: changed \edef to \protected@edef	101
\@glxtrentryfmt: changed \def to \edef to avoid infinite recursion ...	37		
1.44 (2020-03-23)			
\@glxtr@assign@leveloffset: new	146		
\@glxtr@leveloffset: new	146		

<code>\@gls@local@increment@currunitcount:</code>	General: changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> 129	<code>\protected@edef</code> 64, 229
<code>\@gls@setup@default@access:</code>	record: added hybrid 15
changed <code>\edef</code> to	<code>\glossentrydesc:</code> changed <code>\edef</code> to
<code>\protected@edef</code> 193, 194	<code>\protected@edef</code> 208, 209
<code>\@glsxtr@addabbreviationlist:</code>	<code>\Glossentryname:</code> changed <code>\edef</code> to
changed <code>\eappto</code> to	<code>\protected@edef</code> 212, 213
<code>\protected@eappto</code> 136	<code>\glossentryname:</code> changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> 136	<code>\protected@edef</code> 209, 211
<code>\@glsxtr@bibgls@removespaces:</code>	<code>\glossentrynameother:</code> changed <code>\edef</code>
changed <code>\x</code> to <code>\@glo@tmp</code> 444	to <code>\protected@edef</code> 214
<code>\@glsxtr@do@inc@linkcount:</code> changed	<code>\glsadd:</code> changed <code>\edef</code> to
<code>\x</code> to <code>\@glo@tmp</code> 182	<code>\protected@edef</code> 85
<code>\@glsxtr@do@record@wrglossary:</code>	<code>\glsalttreechildpredesc:</code> new 507
changed <code>\edef</code> to <code>\protected@edef</code> . 7	<code>\glsalttreepredesc:</code> new 507
<code>\@glsxtr@do@redef@forglsentries:</code>	<code>\glsdisablehyper:</code> changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> . 5	<code>\protected@edef</code> 109
<code>\@glsxtr@get@prefixedlabel:</code>	<code>\glsdoifexists:</code> changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> 449	<code>\protected@edef</code> 56
changed <code>\x</code> to <code>\@glo@tmp</code> 449	<code>\glsenableentryunitcount:</code> changed
<code>\@glsxtr@mixed@assign@sortkey:</code>	<code>\edef</code> to <code>\protected@edef</code> 131
changed <code>\edef</code> to <code>\protected@edef</code> 147	<code>\glsFindWidestLevelTwo:</code> changed
<code>\@glsxtr@op@recordcounter:</code> changed	<code>\edef</code> to <code>\protected@edef</code> 513
<code>\eappto</code> to <code>\protected@eappto</code> ... 12	<code>\glsFindWidestUsedLevelTwo:</code>
<code>\@glsxtr@orgprintglossary:</code> changed	changed <code>\edef</code> to <code>\protected@edef</code> 512
<code>\xdef</code> to <code>\protected@xdef</code> 143	<code>\glsnavhyperlink:</code> changed <code>\edef</code> to
<code>\@glsxtr@rglstrigger@record:</code>	<code>\protected@edef</code> 106
changed <code>\edef</code> to <code>\protected@edef</code> 177	<code>\glstopicAssignSubIndent:</code> bug 182
<code>\@glsxtr@warn@hybrid@noprintgloss:</code>	maintain hangindent for multiple
new 14	paragraphs 559
<code>\@glsxtrentryfmt:</code> changed <code>\edef</code> to	<code>\glstopicsubitemhangindent:</code> new . 559
<code>\protected@edef</code> 37	<code>\glstopicSubItemParIndent:</code> new .. 559
<code>\@glsxtrglossentry:</code> changed <code>\edef</code> to	<code>\glsxtr@org@newignoredglossary:</code>
<code>\protected@edef</code> 163	changed <code>\eappto</code> to
<code>\@glsxtrglossentryother:</code> changed	<code>\protected@eappto</code> 53
<code>\edef</code> to <code>\protected@edef</code> 164	changed <code>\edef</code> to <code>\protected@edef</code> . 53
<code>\@glsxtrindexaliased:</code> changed <code>\edef</code>	<code>\glsxtr@provideignoredglossary:</code>
to <code>\protected@edef</code> 103	changed <code>\eappto</code> to
<code>\@makeglossaries@warn@noprintglossary:</code>	<code>\protected@eappto</code> 55
new 138	changed <code>\edef</code> to <code>\protected@edef</code> . 54
<code>\@newglossaryentryposthook:</code>	<code>\glsxtr@s@newignoredglossary:</code>
changed <code>\edef</code> to <code>\protected@edef</code> 65	changed <code>\edef</code> to <code>\protected@edef</code> 53
<code>\@print@unsrt@glossary:</code> changed	<code>\glsxtr@s@provideignoredglossary:</code>
<code>\eappto</code> to <code>\protected@eappto</code> .. 167	changed <code>\edef</code> to <code>\protected@edef</code> 55
<code>\@print@unsrt@innerglossary:</code>	<code>\glsxtr@setaccessdisplay:</code> changed
changed <code>\eappto</code> to	<code>\edef</code> to <code>\protected@edef</code> 214
<code>\protected@eappto</code> 169	<code>\glsxtralttreeSymbolDescLocation:</code>
<code>\@printunsrt@glossary@handler:</code>	switch to using <code>\glsalttreepredesc</code>
changed <code>\xdef</code> to <code>\protected@xdef</code> 170	and <code>\glsalttreechildpredesc</code> .. 508

<code>\glstrdisplayendloc</code> : changed <code>\edef</code> to <code>\protected@edef</code>	152	<code>\glstrapptocsvfield</code> : new	44
<code>\glstrdisplaystartloc</code> : changed <code>\edef</code> to <code>\protected@edef</code>	152	<code>\glstrcopytoglossary</code> : replaced <code>\cseappto</code> with <code>\protected@cseappto</code>	55
<code>\glstrdoautoindexname</code> : changed <code>\eappto</code> to <code>\protected@eappto</code> ..	216	<code>\glstrfieldtitlecasescs</code> : added check for <code>\glscapitalisewords</code> ..	207
<code>\glstrseelist</code> : changed <code>\edef</code> to <code>\protected@edef</code>	61	<code>\GlsXtrIfHasNonZeroChildCount</code> : added starred version	439
<code>\glstrtreechildpredesc</code> : new	503	<code>\GlsXtrIfValueInFieldCsvList</code> : new	40
<code>\glstrtreepredesc</code> : new	503	<code>\s@GlsXtrIfFieldNonZero</code> : new	42
<code>\makeglossaries</code> : adjust warning on missing glossary for “alsoindex” ...	139	<code>\s@GlsXtrIfHasNonZeroChildCount</code> : new	439
changed <code>\edef</code> to <code>\protected@edef</code>	139–141	<code>\xGlsXtrIfValueInFieldCsvList</code> : new	40
<code>\makenoidxglossaries</code> : changed <code>\edef</code> to <code>\protected@edef</code>	68	1.48 (2017-02-03) long-postshort-sc-user: new	335
topic: added <code>\par</code> (bug 176)	558	1.48 (2021-11-22) <code>\@@gls@navhypertarget</code> : new	107
grouping added to scope <code>\everypar</code> (bug 182)	558	<code>\@@mgls@hyperlink</code> : new	402
<code>printunsrtglossarywrap</code> : changed <code>\xdef</code> to <code>\protected\xdef</code>	168	<code>\@GlsXtrMglsOrGls</code> : new	427
<code>\setabbreviationstyle</code> : changed <code>\edef</code> to <code>\protected@edef</code>	250	<code>\@Glsfieldorgls</code> : new	433
1.47 () <code>\@GlsXtrIfValueInFieldCsvList</code> : new	40	<code>\@Glsfullorfirst@</code> : new	430
<code>\@xGlsXtrIfValueInFieldCsvList</code> : new	41	<code>\@Glslongortext@</code> : new	430
<code>\s@GlsXtrIfValueInFieldCsvList</code> : new	40	<code>\@Glsshortortext@</code> : new	430
<code>\s@xGlsXtrIfValueInFieldCsvList</code> : new	41	<code>\@PGLSorgls</code> : new	434
1.47 (2017-11-14) <code>\@glstrforcsvfield</code> : new	38	<code>\@PGLSorglspl</code> : new	434
<code>\s@glstrforcsvfield</code> : new	39	<code>\@Pglorgls</code> : new	434
1.47 (2021-11-04) <code>\@GlsXtrIfHasNonZeroChildCount</code> : new	439	<code>\@Pglorglspl</code> : new	434
<code>\@glstrsetaliasnoindex</code> : changed to use <code>\ifcvoid</code>	103	<code>\@alt@GlsXtrMglsOrGls</code> : new	427
<code>\glsaltlistitem</code> : new	490	<code>\@def@multi@glossaryentry</code> : new ..	393
<code>\glslistexpandedname</code> : new	490	<code>\@defmultiglossaryentry</code> : new	392
<code>\glslistgroupafterheader</code> : new ...	491	<code>\@firstofthree</code> : new	408
<code>\glslistinit</code> : new	489	<code>\@gls@combined@category</code> : new	389
<code>\glslistitem</code> : new	489	<code>\@gls@combined@encapmain</code> : new ...	389
<code>\glsseefirstitem</code> : new	62	<code>\@gls@combined@encapothers</code> : new .	389
<code>\glsseelastoxfordsep</code> : new	62	<code>\@gls@combined@firstprefix</code> : new .	390
<code>\glsseelist</code> : redefined	61	<code>\@gls@combined@firstskipmain</code> : new	390
<code>\glsunsetcategoryattribute</code> : new .	202	<code>\@gls@combined@firstskipothers</code> : new	390
		<code>\@gls@combined@firstsuffix</code> : new .	390
		<code>\@gls@combined@hyper</code> : new	388
		<code>\@gls@combined@indexmain</code> : new ...	388
		<code>\@gls@combined@indexothers</code> : new .	388
		<code>\@gls@combined@mglsopts</code> : new	389
		<code>\@gls@combined@mglsopts@do</code> : new	389, 390
		<code>\@gls@combined@mpostlink</code> : new ...	391
		<code>\@gls@combined@mpostlinkelement</code> : new	391

\@gls@combined@postlinks:new ...	391	\@mgls@unsetmain:new	404
\@gls@combined@textformat:new ..	389	\@mgls@unsetothers:new	404
\@gls@combined@usedprefix:new ..	390	\@mglslocalreset:new	398
\@gls@combined@usedskipmain:new	391	\@mglslocalunset:new	398
\@gls@combined@usedskipothers:		\@mglsreset:new	398
new	391	\@mglsunset:new	397
\@gls@combined@usedsuffix:new ..	390	\@multi@glossary@doifexists:new	393
\@gls@do@glsunset:new	102	\@multi@glossary@entry:new	394
\@gls@restore@glslocal:new	102	\@multi@glossaryentry:new	392
\@gls@save@glslocal:new	102	\@multi@glossaryentry@list:new .	394
\@glsfieldorgls:new	433	\@multiglossaryentry:new	392
\@glsfullorfirst@:new	430	\@pglsorgls:new	434
\@glslongortext@:new	430	\@pglsorglspl:new	434
\@glsnavhypertarget:added patch ..	107	\@provide@multi@glossaryentry@noop:	
\@glsshortortext@:new	430	new	394
\@glsshowtarget:new	29	\@secondofthree:new	408
\@glsshowtargetmarkfmt:new	30	\@thirdofthree:new	408
\@glssymbolorgls:new	432	\alt@GlsXtrMglsOrGls:new	427
\@glssymbolorgls:new	432	\glsabbrvsconlyfont:new	361
\@glsxtr@addunused:added check for		\glsabbrvscuserfont:new	334
multientry labels	66	\glscombinedfirstsep:new	421
\@glsxtr@do@org@target:new	147	\glscombinedfirstsepfirst:new ..	421
\@glsxtr@doshowtarget:new	26	\glscombinedsep:new	421
\@glsxtr@mglslike:new	426	\glscombinedsepfirst:new	422
\@glsxtr@mglsrefs:new	425	\glsdoshowtarget:new	29
\@glsxtr@mglswrite:new	424	\glsfirstabbrvsconlyfont:new ...	361
\@glsxtr@multientry:new	396	\glsfirstabbrvscuserfont:new ...	335
\@glsxtr@seefirstitem:new	62	\glslinkwrcontent:new	82
\@glsxtr@seeitem:new	61	\glsnavhypertarget:new	107
\@glsxtrmultientryadjustedname:		\glssetcategoriesattribute:new .	201
new	452	\glssetcategoriesattributes:new	201
\@glsxtrshowtargetleft:new	27	\glssetcombinedsepabbrvnbsp:new	422
\@glsxtrshowtargetmark:new	27	\glssetcombinedsepabbrvnone:new	422
\@glsxtrshowtargetright:new	27	\glssetcombinedsepnarrow:new ...	423
\@mgls@all:new	400	\glsshowtarget:removed check for	
\@mgls@disable@writeseparateref@cond:		\glsshowtargetouter	29
new	425	\glsshowtargetfont:new	29
\@mgls@hyper:new	401	\glsshowtargetinner:new	29
\@mgls@hyperlink:new	402	\glsshowtargetinnercontentsymleft:	
\@mgls@main:new	401	new	29
\@mgls@others:new	401	\glsshowtargetinnercontentsymright:	
\@mgls@resetall:new	403	new	29
\@mgls@resetmain:new	403	\glsshowtargetouter:new	29
\@mgls@resetothers:new	403	\glsxtr@mgls@applyopts:new	407
\@mgls@setup:new	401	\glsxtr@mgls@checklastelement:	
\@mgls@setup@do:new	401	new	406
\@mgls@setup@do@not:new	401	\glsxtr@mgls@inner:new	408
\@mgls@unsetaction:new	401	\glsxtr@newmgls:new	426
\@mgls@unsetall:new	404	\glsxtr@setup@docurrent:new	405

\glxtrdohyperlink: added check for multi-entry	108	\glxtrshowtargetouter: new	27
\glxtrifmulti: new	391	\glxtrshowtargetsymbolleft: new .	29
\glxtrlongshortscuserdescname: new	337	\glxtrshowtargetsymboloright: new	29
\GlsXtrMglsOrGls: new	426	showtargets: new	27
\glxtrmglsWarnAllSkipped: new ..	406	\if@mgl@s@writeseparaterefs: new .	425
\GLSxtrmultientryadjustedname: new	452	\ifKV@mgl@s@presetlocal: new	401
\GlsXtrmultientryadjustedname: new	451	\ifmglsused: new	397
\Glsxtrmultientryadjustedname: new	451	\ifmultiglossaryentryglobal: new	392
\glxtrmultientryadjustedname: new	451	\MGLS: new	429
\GLSxtrmultientryadjustednamefmt: new	453	\MGLs: new	429
\GlsXtrmultientryadjustednamefmt: new	453	\Mgls: new	428
\Glsxtrmultientryadjustednamefmt: new	453	\mgls: new	427
\GLSxtrmultientryadjustednamefmt: new	453	\mgls@disable@mglsopts: new	390
\glxtrmultientryadjustednamefmt: new	453	\mgls@disable@setup: new	401
\GLSxtrmultientryadjustednameother: new	453	\mgls@enable@mglsopts: new	390
\GlsXtrmultientryadjustednameother: new	453	\mgls@enable@setup: new	401
\Glsxtrmultientryadjustednameother: new	453	\mglsAddOptions: new	400
\glxtrmultientryadjustednameother: new	453	\mglscustompostlinkhook: new	419
\GLSxtrmultientryadjustednameother: new	453	\mglsdefcategoryprefix: new	420
\GlsXtrmultientryadjustednameother: new	453	\mglsdefcategorysuffix: new	420
\Glsxtrmultientryadjustednameother: new	453	\mglsselementindex: new	392
\glxtrmultientryadjustednameother: new	453	\mglsselementposthook: new	421
\GLSxtrmultientryadjustednameother: new	453	\mglsselementprehook: new	421
\GlsXtrmultientryadjustednameother: new	453	\mglsselementreset: new	403
\Glsxtrmultientryadjustednameother: new	453	\mglsselementunset: new	403
\glxtrmultientryadjustednamepostsep: new	453	\mglsfield: new	433
\glxtrmultientryadjustednamepresep: new	453	\mglsforelements: new	402
\glxtrmultientryadjustednamesep: new	453	\mglsforotherelements: new	402
\glxtrmultilastotherindex: new .	391	\Mglsfull: new	431
\glxtrmultilist: new	391	\mglsfull: new	431
\glxtrmultimain: new	391	\mglsfull: new	431
\glxtrmultimainindex: new	391	\mglsfull: new	431
\glxtrmultitotalelements: new ..	391	\mglsfull: new	431
\glxtrsconlydescname: new	363	\mglsfull: new	431
\glxtrsconlydescsort: new	363	\mglsfull: new	431
\glxtrsconlyname: new	362	\mglsfull: new	431
\glxtrsconlysuffix: new	362	\mglsfull: new	431
\glxtrsconlyusername: new	335	\mglsfull: new	431
\glxtrsconlysuffix: new	335	\mglsfull: new	431
\glxtrshowtargetinner: new	27	\mglsfull: new	431

\mglspl:new	428	\mglsWriteSeparateRefsTrue:new .	425
\mglsprefix:new	420	\MPGLS:new	436
\mglsreset:new	397	\MPGLs:new	436
\mglsresetall:new	399	\Mpgls:new	435
\mglsseefirstitem:new	62	\mpgls:new	435
\mglsseeitem:new	62	\MPGLSmainpl:new	436
\mglsSetMain:new	399	\MPGLsmainpl:new	436
\mglsSetOptions:new	400	\Mpglsmainpl:new	435
\Mglsshort:new	431	\mpglsmainpl:new	435
\mglsshort:new	430	\MPGLSpl:new	436
\mglsuffix:new	421	\MPGLspl:new	436
\MGlssymbol:new	433	\Mpglspl:new	435
\Mglssymbol:new	432	\mpglspl:new	435
\mglsymbol:new	432	\mpglsWarning:new	434
\mglsunset:new	397	\multiglossaryentry:new	392
\mglsunsetall:new	398	\multiglossaryentrysetup:new ...	388
\mglsunsetothers:new	402	long-only-short-sc-only:new	362
\mglsusecategoryprefix:new	420	long-only-short-sc-only-desc:new	363
\mglsusecategorysuffix:new	421	long-postshort-sc-user-desc:new	338
\MGlseusefield:new	433	\p@GlsXtrMglsOrGls:new	427
\Mglseusefield:new	433	\providemultiglossaryentry:new .	393
\mglsusefield:new	433	\s@GlsXtrMglsOrGls:new	427
\mglsWriteSeparateRefsFalse:new	425	\writemultiglossentry:new	397

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	
\\$	439
\,	60
\.	18, 19, 225, 500
\@	67, 159, 192, 232
\@@cGLS@	123, 131
\@@cGLSpl@	123, 131
\@@cGLs@	123, 131
\@@cGLspl@	123, 131
\@@cgls@	123, 131
\@@cglspl@	123, 131
\@@do@wrglossary	7, 9, 140
\@@do@wrglossary	12, 14–16, 31, 86, 103
\@@glo@assign@sortkey	147
\@@glo@list	5
\@@glo@type	165
\@@glossarysec	528
\@@glossaryseclabel	146
\@@gls@expand@field	35
\@@gls@navhypertarget	106, 107
\@@glslocalreset	120
\@@glslocalunset	120
\@@glsreset	120
\@@glsunset	119
\@@glsxtr@autoindex@escspch	218–220
\@@glsxtr@base@acrcmd@warn	94, 136
\@@glsxtr@checkspch	217, 218, 220
\@@glsxtr@disabledflycommand	72, 73
\@@glsxtr@org@postdescription	144
\@@glsxtr@record	14, 16
\@@glsxtr@recordcounter	14–16, 162
\@@glsxtrfmt	35, 36
\@@glsxtrp	114
\@@glsxtrpostloctag	75
\@@glsxtrpreloctag	75
\@@glsxtrwrglossmark	7, 8, 12, 30, 31, 63, 68, 140
\@@mgls@hyperlink	402
\@@newglossaryentry@defcounters ...	122
\@@newglossaryentry@defunitcounters	130
\@@par	508
\@ACRlong	110
\@ACRlongpl	110
\@ACRshort	110
\@ACRshortpl	110
\@Acrlong	110
\@Acrlongpl	110
\@Acrshort	110
\@Acrshortpl	110
\@GLS@	110, 125, 126, 180, 429, 434, 436, 450
\@GLSdesc@	91
\@GLSpl@ .	110, 126, 127, 181, 429, 434, 436, 450
\@GLSplural@	111
\@GLSsymbol@	92
\@GLStext@	111
\@GLSxtr@full	237
\@GLSxtr@fullpl	238
\@GLSxtr@p@acrlong@	110
\@GLSxtr@p@acrlongpl@	110
\@GLSxtr@p@acrshort@	110
\@GLSxtr@p@acrshortpl@	110
\@GLSxtr@p@long@	110
\@GLSxtr@p@longpl@	110
\@GLSxtr@p@plural@	110
\@GLSxtr@p@short@	110
\@GLSxtr@p@shortpl@	110
\@GLSxtr@p@text@	110
\@GLSxtrlong	110, 242
\@GLSxtrlongpl	110, 246
\@GLSxtrp	117, 118
\@GLSxtrshort	110, 240
\@GLSxtrshortpl	110, 244
\@Gls@	110, 125, 126, 179, 428, 429, 432–434, 436, 450

\@Gls@acrentryname	134	\@Pgls@	434
\@Gls@entry@field	44, 99, 116, 117, 215	\@Pglsorgls@	435, 436
\@Gls@entryname	134	\@Pglsorglspl@	435, 436
\@GlsXtrEnableOnTheFly	69, 70	\@Pglspl@	434
\@GlsXtrIfFieldCmpNum	42	\@acrlong	110
\@GlsXtrIfFieldEqNum	42	\@acrlongpl	110
\@GlsXtrIfFieldEqStr	46	\@acrshort	110
\@GlsXtrIfFieldEqXpStr	46	\@acrshortpl	110
\@GlsXtrIfFieldNonZero	42, 439	\@addtoacronymlists	134–137
\@GlsXtrIfFieldValueInCsvList	39	\@addtoreset	182
\@GlsXtrIfHasNonZeroChildCount	439	\@afterheading	490,
\@GlsXtrIfValueInFieldCsvList	40		491, 502, 503, 505, 507, 520–524, 532, 558
\@GlsXtrIfXpFieldEqXpStr	47	\@alt@GlsXtrMglsOrGls	427
\@GlsXtrMglsOrGls	427	\@alt@gls@hyp@opt	105
\@GlsXtrStartUnsetBuffering	119	\@auxout	10–12, 68, 76,
\@GlsXtrStopUnsetBuffering	119		106, 107, 123, 124, 132, 139, 140, 154,
\@Glsfieldorgls	433, 434		159–161, 163, 174, 397, 425, 426, 449, 532
\@Glsfirst@	430	\@bibgls@restoreat	159
\@Glsfullorfirst	430, 431	\@cGLS	126
\@Glslongortext	430, 431	\@cGLS@	123, 126, 131
\@Glsname@	432	\@cGLSpl	126
\@Glspl@	110, 125, 126, 180, 428, 429, 434, 436, 450	\@cGLSpl@	123, 127, 131
\@Glsplural@	111	\@cGls@	123, 131
\@Glsshortortext	430, 431	\@cGlspl@	123, 131
\@Glstext@	110, 430	\@cglspl@	123, 131
\@Glsxtr	71, 72	\@currentlabelname	146
\@Glsxtr@full	236, 430	\@dGLS	450
\@Glsxtr@fullpl	238	\@dGLSpl	450
\@Glsxtr@p@acrlong@	110	\@dGls	449
\@Glsxtr@p@acrlongpl@	110	\@dGlspl	450
\@Glsxtr@p@acrshort@	110	\@dblfloat	18
\@Glsxtr@p@acrshortpl@	110	\@def@multi@glossaryentry	392, 393
\@Glsxtr@p@long@	110	\@def@multi@glossaryentry@do	393
\@Glsxtr@p@longpl@	110	\@defmultiglossaryentry	392
\@Glsxtr@p@plural@	110	\@dgls	449
\@Glsxtr@p@short@	110	\@dglspl	449
\@Glsxtr@p@shortpl@	110	\@disable@onlypremakeg	140
\@Glsxtr@p@text@	110	\@do@auxoutstuff	154
\@Glsxtrlong	110, 242, 430	\@do@gls@getcounterprefix	10, 11
\@Glsxtrlongpl	110, 245	\@do@glssee	64, 65
\@Glsxtrp	116, 117	\@do@newglossaryentry	134, 135, 233
\@Glsxtrpl	72	\@do@seeglossary	14–16, 30, 68, 140
\@Glsxtrshort	110, 240, 430	\@do@wrglossary	84, 178
\@Glsxtrshortpl	110, 243	\@domakeglossaries	68, 139
\@PGLS@	434	\@dtl@formatlist@handler	38
\@PGLSorgls@	436	\@dtl@formatlist@itemsep	38
\@PGLSorglspl@	436	\@dtl@formatlist@lastitem	38
\@PGLSpl@	434	\@dtl@formatlist@prelastitem	38

<code>\@dtl@formatlist@prelastitemsep</code>	38	<code>\@glo@type</code> 66, 106, 107, 134, 136, 139, 143,	
<code>\@dtlformatlist</code>	39	144, 147, 150, 151, 154, 157, 158, 166–169	
<code>\@empty</code> ..	32, 86, 94–98, 136, 145, 152, 217,	<code>\@glo@types</code>	14, 138, 205, 510–517
218, 236–246, 395, 407–409, 411, 413, 418		<code>\@glossary@default@style</code>	73, 143, 167, 526
<code>\@end@glxtr@addunused</code>	66	<code>\@glossarystyle</code>	143, 167, 168
<code>\@end@glxtr@gettype</code>	142, 147	<code>\@glossentrysymbol</code>	220, 221
<code>\@end@glxtr@usesee</code>	58	<code>\@gls@</code> ...	110, 124, 126, 179, 428, 432–435, 449
<code>\@end@glxtr@trifhyphenstart</code>	344	<code>\@gls@@automake@immediate</code>	139
<code>\@endfortrue</code>	39, 214, 250, 449	<code>\@gls@@link</code>	79
<code>\@firstofone</code>		<code>\@gls@ReturnAfterFi</code>	153, 444
.....	86, 166, 169, 192, 208, 209, 216,	<code>\@gls@actualchar</code>	217
221, 222, 390, 401, 405, 406, 410, 443, 489		<code>\@gls@actuallong</code>	193, 194
<code>\@firstofthree</code> ..	79, 86, 94–97, 105, 236,	<code>\@gls@actuallongpl</code>	193, 194
238, 239, 241, 243, 245, 408, 411, 418, 419		<code>\@gls@actualshort</code>	193, 194
<code>\@firstoftwo</code>	87–	<code>\@gls@actualshortpl</code>	193, 194
92, 95–98, 101, 105, 137, 172, 214, 227,		<code>\@gls@adjustmode</code>	85
228, 236, 238, 239, 243–246, 366, 367,		<code>\@gls@alt@hyp@opt</code>	105
406, 408, 409, 411, 413, 415, 418, 419, 427		<code>\@gls@alt@hyp@opt@char</code>	105, 106, 427
<code>\@float</code>	18	<code>\@gls@alt@hyp@opt@keys</code>	105, 106
<code>\@for</code>	5, 24, 39, 61,	<code>\@gls@assign@actual</code>	193
66, 86, 121, 122, 133, 136, 137, 139, 142,		<code>\@gls@automake</code>	142
150, 166, 169, 177, 183, 201, 207, 214,		<code>\@gls@between</code>	150
222, 394, 398, 399, 402–405, 413, 449, 452		<code>\@gls@checkedmkidx</code>	217, 218, 220
<code>\@glo@alias</code>	63, 64	<code>\@gls@checkmkidxchars</code>	63, 217
<code>\@glo@assign@sortkey</code>	142	<code>\@gls@codepage</code>	154
<code>\@glo@autosee</code>	30	<code>\@gls@combined@category</code>	409, 410
<code>\@glo@autoseehook</code>	64	<code>\@gls@combined@current@label</code> ..	392–397
<code>\@glo@category</code>	128	<code>\@gls@combined@current@main</code>	394, 395, 399
<code>\@glo@check@sortallowed</code>	142	<code>\@gls@combined@current@opts</code>	
<code>\@glo@counterprefix</code>	10, 11, 32, 152, 153, 444	392, 393, 395–397
<code>\@glo@countunit</code>	128	<code>\@gls@combined@currentlist</code> .	394–396, 399
<code>\@glo@default@sorttype</code>	142	<code>\@gls@combined@encapmain</code>	415
<code>\@glo@desc</code>	52	<code>\@gls@combined@encapothers</code>	416
<code>\@glo@descplural</code>	52	<code>\@gls@combined@encapsulator</code> ...	410, 413
<code>\@glo@group</code>	13	<code>\@gls@combined@finalitem</code>	394, 395, 399, 400
<code>\@glo@label</code>	13, 34, 57, 64–66, 99, 109, 510–517	<code>\@gls@combined@firstprefix</code>	411
<code>\@glo@location</code>	13	<code>\@gls@combined@firstsuffix</code>	411
<code>\@glo@loclist</code>	13	<code>\@gls@combined@hyper</code>	410, 415, 416
<code>\@glo@name</code>	216, 217	<code>\@gls@combined@hyper@val</code>	389
<code>\@glo@no@assign@sortkey</code>	147	<code>\@gls@combined@indexmain</code>	415
<code>\@glo@parent</code>	512, 513	<code>\@gls@combined@indexmain@val</code>	388
<code>\@glo@see</code>	57, 58, 60, 61, 64–66	<code>\@gls@combined@indexothers</code>	416
<code>\@glo@seealso</code>	63, 64	<code>\@gls@combined@indexothers@val</code>	388
<code>\@glo@sort</code>	217	<code>\@gls@combined@mglsopts</code>	409
<code>\@glo@sorttype</code>	142, 151	<code>\@gls@combined@mglsopts@do</code>	389, 390
<code>\@glo@text</code>	79	<code>\@gls@combined@mglsopts@do@not</code>	390
<code>\@glo@thislettergrp</code>	172	<code>\@gls@combined@mpostlink@nr</code>	391, 411, 412
<code>\@glo@thisvalue</code>	331	<code>\@gls@combined@mpostlink@val</code>	391
<code>\@glo@tmp</code> ...	32, 34, 61, 99, 153, 182, 444, 449		

\@gls@combined@postlinkelement@nr	\@gls@link@nocheckfirsthyper
..... 391, 411, 412 78, 94–98, 236–246
\@gls@combined@postlinkelement@val	\@gls@link@opts
391 82
\@gls@combined@postlinks@nr	\@gls@list
... 391, 413 107, 150
\@gls@combined@postlinks@val	\@gls@local@increment@currcount
..... 391	... 123
\@gls@combined@textformat	\@gls@local@increment@currunitcount
..... 410	131
\@gls@combined@usedprefix	\@gls@location
..... 411 172, 173
\@gls@combined@usedsuffix	\@gls@loclist
..... 411 147–149, 172, 173
\@gls@counter	\@gls@long
..... 8, 10, 11, 26, 32, 81, 82, 85, 103, 177 231
\@gls@currentlettergroup	\@gls@longaccess
151, 166, 169, 172 191, 193, 194
\@gls@declareoption	\@gls@longaccesspl
..... 4 191, 194
\@gls@default@longpl	\@gls@longpl
..... 231, 232 193, 229, 231–233
\@gls@deffile	\@gls@map
..... 67 214
\@gls@do@glssunset	\@gls@nameaccess
..... 79, 102, 178 191, 194
\@gls@doautomake	\@gls@nohyperlist
..... 142, 162 53, 55
\@gls@doautomake@err	\@gls@noidx@do
..... 162 151
\@gls@dolast	\@gls@noidx@getgrouptitle
..... 61 166, 169
\@gls@donext	\@gls@noidx@nosanitizesort
..... 61 141
\@gls@enablesavenonumberlist	\@gls@noidx@sanitizesort
..... 67 141
\@gls@encapchar	\@gls@noidx@loclist@finalsep
..... 218	... 147, 148
\@gls@entry@count	\@gls@noidx@loclist@prev
..... 123, 124 148
\@gls@entry@field	\@gls@noidx@loclist@sep
..... 35, 43, 44, 64, 99, 115–118, 122, 164, 165 147
\@gls@entry@unitcount	\@gls@noref@warn
..... 131, 132 140, 151
\@gls@field@font	\@gls@org@combined@encapsulator
..... 86–94	410, 411
\@gls@field@link	\@gls@org@glsnoidxdisplayloc
..... 87–94, 100, 101 148
\@gls@firstaccess	\@gls@org@glssseeformat
..... 191, 195 148
\@gls@firstpluralaccess	\@gls@org@longpl
..... 191, 195 232
\@gls@get@counterprefix	\@gls@org@shortpl
..... 32 232
\@gls@getcounterprefix	\@gls@pluralaccess
..... 10 191, 195
\@gls@getgrouptitle	\@gls@preglossaryhook
.... 149, 150, 166, 169 144, 168, 222
\@gls@grptitle	\@gls@prevlevel
..... 106, 150 518–520, 524–526
\@gls@hyp@opt	\@gls@quotechar
..... 100, 101, 105, 126, 174, 178–180, 236–246, 426, 449, 450 217
\@gls@hyp@opt@cs	\@gls@reference
..... 105 68, 139, 140
\@gls@hypergroup	\@gls@resetlabel
..... 107 403, 404
\@gls@ifaccessattribute@set	\@gls@restore@glsllocal
... 193, 194 78, 85, 178
\@gls@ifinlist	\@gls@restoreat
..... 170 67
\@gls@increment@currcount	\@gls@save@glsllocal
..... 122 78, 82, 178
\@gls@increment@currunitcount	\@gls@saveentrycounter
..... 131	14–16, 32, 83, 85, 178
\@gls@initaccesskeys	\@gls@see@noindex
..... 231 31, 159
\@gls@keymap	\@gls@setdefault@glslink@opts
..... 13, 34, 57, 63, 99, 160, 214 8, 36, 83, 104
\@gls@label	\@gls@setsort
..... 7, 8, 10, 11, 68, 104, 105, 140, 163, 250 83, 85
\@gls@levelchar	\@gls@setup@default@access
..... 218 233
\@gls@link	\@gls@setupsort@none
..... 36, 78, 79, 94–98, 236–246 15
\@gls@link@checkfirsthyper	\@gls@short
..... 79, 137 232
\@gls@link@label	\@gls@shortaccess
..... 82, 177 191, 193–195
	\@gls@shortaccesspl
 191, 194, 195
	\@gls@shortpl
 193, 229, 232, 233
	\@gls@sort
 172

<code>\@gls@textaccess</code>	191, 195	<code>\@glsshowtargetmarkfmt</code>	28
<code>\@gls@thisHloc</code>	32	<code>\@glsstyle@altlist</code>	490
<code>\@gls@thisattrlabel</code>	201	<code>\@glsstyle@altlistgroup</code>	491
<code>\@gls@thiscatlabel</code>	201	<code>\@glsstyle@altlisthypergroup</code>	492
<code>\@gls@thishypernavlabel</code>	107	<code>\@glsstyle@alttree</code>	507
<code>\@gls@thislabel</code>	61, 86, 121, 449, 450	<code>\@glsstyle@alttreegroup</code>	519
<code>\@gls@thisloc</code>	32	<code>\@glsstyle@alttreehypergroup</code>	520
<code>\@gls@thisval</code>	214	<code>\@glsstyle@index</code>	501
<code>\@gls@tmp</code> .. 41, 46, 47, 150, 193, 394, 399, 402		<code>\@glsstyle@indexgroup</code>	502
<code>\@gls@tmpb</code>	220	<code>\@glsstyle@indexhypergroup</code>	502
<code>\@gls@type</code> .. 136, 137, 140–142, 250, 510–517		<code>\@glsstyle@inline</code>	500
<code>\@gls@unsetlabel</code>	404, 405	<code>\@glsstyle@list</code>	489
<code>\@gls@write@entrycounts</code>	123	<code>\@glsstyle@listdotted</code>	488
<code>\@gls@write@entryunitcounts</code>	131	<code>\@glsstyle@listgroup</code>	491
<code>\@gls@write@entryunitcounts@do</code>	132	<code>\@glsstyle@listhypergroup</code>	491
<code>\@gls@writedef</code>	67, 68	<code>\@glsstyle@mcolalttree</code>	524
<code>\@gls@xref</code>	12, 63	<code>\@glsstyle@mcolalttreegroup</code>	524
<code>\@gls@extradjustedother</code>	452	<code>\@glsstyle@mcolalttreehypergroup</code> ..	525
<code>\@gls@xtrmultientryadjustednamesep</code>	452, 453	<code>\@glsstyle@mcolalttreespannav</code>	525
<code>\@glsabbrv@current@abbreviation</code> 231, 247		<code>\@glsstyle@mcolindexgroup</code>	520
<code>\@glsacronymlists</code>	134–137	<code>\@glsstyle@mcolindexhypergroup</code>	521
<code>\@glsdisp</code>	433	<code>\@glsstyle@mcolindexspannav</code>	521
<code>\@glsdoifexistsorwarn</code> 17, 209, 211, 212, 214		<code>\@glsstyle@mcoltreegroup</code>	521
<code>\@glsentry</code>	67, 123, 124, 132	<code>\@glsstyle@mcoltreehypergroup</code>	522
<code>\@glsfieldorgls</code>	433	<code>\@glsstyle@mcoltreenonamegroup</code>	523
<code>\@glsfirst@</code>	430	<code>\@glsstyle@mcoltreenonamehypergroup</code> 523	
<code>\@glsfullorfirst</code>	430, 431	<code>\@glsstyle@mcoltreenonamespannav</code> ..	523
<code>\@glslink</code>	84, 106, 109, 402	<code>\@glsstyle@mcoltreespannav</code>	522
<code>\@glslocalreset</code>	121	<code>\@glsstyle@tree</code>	503
<code>\@glslocalunset</code>	120, 121	<code>\@glsstyle@treegroup</code>	505
<code>\@glslongextra@begintab</code> 538–552, 554–556		<code>\@glsstyle@treehypergroup</code>	505
<code>\@glslongextrawidestname</code>	536, 537	<code>\@glsstyle@treenoname</code>	505
<code>\@glslongortext</code>	430, 431	<code>\@glsstyle@treenonamegroup</code>	507
<code>\@glsname@</code>	431, 432	<code>\@glsstyle@treenonamehypergroup</code> ...	507
<code>\@glsnavhypertarget</code>	107	<code>\@glssymbol@</code>	432
<code>\@glsnextpages</code>	143, 168	<code>\@glssymbolorgls</code>	432, 433
<code>\@glsnonextpages</code>	143, 168	<code>\@glssymbolorgls</code>	432
<code>\@glsnumberformat</code>	8, 10, 11, 82, 85, 103, 177, 213, 216, 443, 444	<code>\@glstarget</code>	109, 145–147
<code>\@glsorder</code>	139	<code>\@glstext@</code>	110, 430
<code>\@glspl@</code> . 110, 125, 126, 179, 428, 434, 435, 449		<code>\@glunset</code>	119, 120
<code>\@glsplural@</code>	111	<code>\@glswidestname</code>	510, 518
<code>\@gls punc@token</code>	228	<code>\@glsextr@newmgls@do</code>	426
<code>\@glsrecordlocref</code>	10	<code>\@glsextr</code>	70, 72
<code>\@glsseeitem</code>	61	<code>\@glsextr@@do@@wrglossary</code>	140
<code>\@glsseelastsep</code>	61	<code>\@glsextr@abbreviationsdef</code>	20, 31
<code>\@glsshortortext</code>	430, 431	<code>\@glsextr@abbrlists</code>	136
<code>\@glsshowtarget</code>	27	<code>\@glsextr@accessdisplay</code>	214, 215
		<code>\@glsextr@acronymlists</code>	136, 137

<code>\@glsxtr@activate@initialtagging</code> ..	<code>\@glsxtr@declareoption</code> ... 4, 18, 20, 23, 26
..... 222, 223	<code>\@glsxtr@defaultnoglossarywarning</code> .. 24
<code>\@glsxtr@addabbreviationlist</code> 233	<code>\@glsxtr@defaultnumberformat</code>
<code>\@glsxtr@addunitcounter</code> 128 6, 8, 82, 85, 103, 213, 216
<code>\@glsxtr@addunused</code> 66	<code>\@glsxtr@defpostpunc</code> 18, 19, 28
<code>\@glsxtr@addunusedxrefs</code> 66	<code>\@glsxtr@deprecated@abbrstyle</code>
<code>\@glsxtr@altmodifier</code> 106 282, 284, 285,
<code>\@glsxtr@assign@leveloffset</code> 146	288, 299, 300, 302, 305, 319, 323, 327, 329
<code>\@glsxtr@attrval</code> 84, 208–214, 216, 221	<code>\@glsxtr@dialect</code> 48, 49
<code>\@glsxtr@autoindex@at</code> 217, 218	<code>\@glsxtr@disabledflycommand</code> 72
<code>\@glsxtr@autoindex@doextra@esc</code> 217	<code>\@glsxtr@display@loc</code> 151
<code>\@glsxtr@autoindex@encap</code> ... 216, 218, 219	<code>\@glsxtr@do@@wrindex</code> 104, 105
<code>\@glsxtr@autoindex@esc</code> 217, 219, 220	<code>\@glsxtr@do@autoadd</code> 82
<code>\@glsxtr@autoindex@escat</code> 217, 218	<code>\@glsxtr@do@glsgldisablehyperinlist</code> . 102
<code>\@glsxtr@autoindex@escencap</code> ... 218, 219	<code>\@glsxtr@do@inc@linkcount</code> 183
<code>\@glsxtr@autoindex@esclevel</code> ... 218, 219	<code>\@glsxtr@do@nameref@record</code> 10, 11
<code>\@glsxtr@autoindex@escquote</code> ... 217, 219	<code>\@glsxtr@do@org@target</code> 107
<code>\@glsxtr@autoindex@level</code> 218, 219	<code>\@glsxtr@do@record@wrglossary</code> 7, 14
<code>\@glsxtr@autoindex@setname</code> 216	<code>\@glsxtr@do@redef@forglsentries</code> 6
<code>\@glsxtr@autoindexcrossrefs</code> 14, 17, 57, 64	<code>\@glsxtr@do@style</code> 25, 437
<code>\@glsxtr@autoseeindexfalse</code> 14	<code>\@glsxtr@do@titlecaps@warn</code>
<code>\@glsxtr@autoseeindextrue</code> 17 208–211, 215, 223
<code>\@glsxtr@base@acrcmd</code> 94–99, 136, 137	<code>\@glsxtr@doabbreviationsdef</code> 20
<code>\@glsxtr@bibgls@removespaces</code> 444	<code>\@glsxtr@doaccsupp</code> 23, 28
<code>\@glsxtr@bookindex@atendgroup</code>	<code>\@glsxtr@docdefs@multilist</code> 393
..... 529, 530, 532	<code>\@glsxtr@docdefsetting</code> 17, 67
<code>\@glsxtr@bookindex@atsubendgroup</code> .. 531	<code>\@glsxtr@docdefval</code>
<code>\@glsxtr@bookindex@atsubsubendgroup</code> 531 16, 17, 67, 69, 392, 393, 396
<code>\@glsxtr@bookindex@between</code> . 529, 530, 532	<code>\@glsxtr@doccounterrecord</code> 12
<code>\@glsxtr@bookindex@sep</code> 529–531	<code>\@glsxtr@doglossary</code> 166, 167, 169
<code>\@glsxtr@bookindex@subatendgroup</code> ..	<code>\@glsxtr@doiflabelinlist</code> 170
..... 529, 530, 532	<code>\@glsxtr@doloadprefix</code> 23, 28, 31
<code>\@glsxtr@bookindex@subbetween</code> . 529–531	<code>\@glsxtr@doloctag</code> 75, 76
<code>\@glsxtr@bookindex@subsep</code> 529–531	<code>\@glsxtr@dorecord</code> 7, 9
<code>\@glsxtr@bookindex@subsubatendgroup</code>	<code>\@glsxtr@dorecordnodefer</code> 7, 9
..... 530, 532	<code>\@glsxtr@dosee@alsoindex@glossary</code> .. 16
<code>\@glsxtr@bookindex@subsubbetween</code> ..	<code>\@glsxtr@doseeglossary</code> 15, 30
..... 529–531	<code>\@glsxtr@doshowtarget</code> 27–29
<code>\@glsxtr@bookindexgroupskip</code> ... 530, 532	<code>\@glsxtr@dostylewarn</code> 250
<code>\@glsxtr@cat</code> 122, 133, 177, 222	<code>\@glsxtr@enabletagging</code> 222
<code>\@glsxtr@check@bibgls@nameref</code> 159	<code>\@glsxtr@end@</code>
<code>\@glsxtr@checkgroup</code> 166, 169 70
<code>\@glsxtr@counterrecordhook</code> 10–12	<code>\@glsxtr@endescspch</code> 217–220
<code>\@glsxtr@csname</code> 128, 129, 131	<code>\@glsxtr@entrycount@org@localreset</code> 123
<code>\@glsxtr@current@style</code> 73, 526	<code>\@glsxtr@entrycount@org@localunset</code> 123
<code>\@glsxtr@currentunitcount</code> .. 128, 129, 131	<code>\@glsxtr@entrycount@org@reset</code> 123
<code>\@glsxtr@currunitcount</code> 130, 132	<code>\@glsxtr@entrycount@org@unset</code> 122
<code>\@glsxtr@debugnr</code> 27	<code>\@glsxtr@entryunitcount@org@localreset</code>
<code>\@glsxtr@debugval</code> 27 131

\@glsxtr@entryunitcount@org@localunset	\@glsxtr@mglrefs	425
..... 131	\@glsxtr@mixed@assign@sortkey	142
\@glsxtr@entryunitcount@org@reset	\@glsxtr@multientry	397
..... 131	\@glsxtr@multilabel	66
\@glsxtr@entryunitcount@org@unset	\@glsxtr@newglslike	174
..... 130, 131	\@glsxtr@noidx@displaynumberlist	141
\@glsxtr@equationsfalse	\@glsxtr@noidx@do	171
..... 18	\@glsxtr@noidx@entrynumberlist	141
\@glsxtr@err@undefaction	\@glsxtr@noidx@numberlistloop	141
..... 6, 15	\@glsxtr@nomissingglsstextnr	24
\@glsxtr@field@linkdefs	\@glsxtr@nomissingglsstextval	24
..... 78, 79	\@glsxtr@noop@recordcounter	12, 15
\@glsxtr@floatsfalse	\@glsxtr@nopostpunc	144
..... 18	\@glsxtr@nopostpunc@postdesc	144, 145
\@glsxtr@format@overridefalse	\@glsxtr@notfoundinlist	228
..... 215	\@glsxtr@op@recordcounter	14, 16
\@glsxtr@format@overridetrue	\@glsxtr@optlist	72
..... 216	\@glsxtr@org@@starttoc	365
\@glsxtr@foundinlist	\@glsxtr@org@GLS@	79
..... 228	\@glsxtr@org@GLSpl@	79
\@glsxtr@full	\@glsxtr@org@Gls@	78
..... 236, 430	\@glsxtr@org@Glspl@	79
\@glsxtr@fullpl	\@glsxtr@org@Glsxtrtitlefirst	366, 368
..... 237	\@glsxtr@org@Glsxtrtitlefirstplural	366, 368
\@glsxtr@get@prefixedlabel	\@glsxtr@org@Glsxtrtitlefull	367, 368
..... 449, 450	\@glsxtr@org@Glsxtrtitlefullpl	367, 368
\@glsxtr@gettype	\@glsxtr@org@Glsxtrtitlelong	366, 368
..... 142	\@glsxtr@org@Glsxtrtitlelongpl	366, 368
\@glsxtr@glossdescfont	\@glsxtr@org@Glsxtrtitlename	366, 367
..... 208, 209	\@glsxtr@org@Glsxtrtitleplural	366, 367
\@glsxtr@glossnamefont	\@glsxtr@org@Glsxtrtitleshort	366, 367
..... 209–215	\@glsxtr@org@Glsxtrtitleshortpl	366, 367
\@glsxtr@glosssymbolfont	\@glsxtr@org@Glsxtrtitletext	366, 367
..... 221	\@glsxtr@org@MakeUppercase	366, 367
\@glsxtr@gobbleto@endescspch	\@glsxtr@org@addtoacronymlists	134, 137
..... 220	\@glsxtr@org@checkfirsthyper	101, 137
\@glsxtr@groupheading	\@glsxtr@org@currentfieldvalue	47, 48
..... 166, 169, 172	\@glsxtr@org@delimN	75, 76
\@glsxtr@idx@displaynumberlist	\@glsxtr@org@delimR	75, 76
..... 141	\@glsxtr@org@doseeglossary	30, 140
\@glsxtr@idx@entrynumberlist	\@glsxtr@org@gloautosee	30, 31
..... 141	\@glsxtr@org@glolinkprefix	82, 84
\@glsxtr@if@record@only	\@glsxtr@org@gls@	78
..... 139, 162	\@glsxtr@org@glsdohypertarget	146, 147
\@glsxtr@ifcsstart	\@glsxtr@org@glsignore	75, 76
..... 70	\@glsxtr@org@glspl@	78
\@glsxtr@ifischild	\@glsxtr@org@Glsxtrtitlefirst	366, 368
..... 172, 173		
\@glsxtr@ifnum@mmode		
..... 81		
\@glsxtr@ifpunctoken		
..... 228		
\@glsxtr@ifunitcounter		
..... 128		
\@glsxtr@insert@dots		
..... 229		
\@glsxtr@insert@dots@next		
..... 230		
\@glsxtr@insert@dots		
..... 193, 232		
\@glsxtr@label		
..... 39, 66, 183, 207		
\@glsxtr@labellist		
..... 66		
\@glsxtr@labelprefixes		
..... 448, 449		
\@glsxtr@leveloffset		
..... 146, 172		
\@glsxtr@loadstyles		
..... 487, 488		
\@glsxtr@local@textformat		
..... 82–84		
\@glsxtr@locale		
..... 48, 49		
\@glsxtr@longnewglossaryentry		
..... 51		
\@glsxtr@mark@wordseps		
..... 230		
\@glsxtr@mark@wordseps@next		
..... 230		
\@glsxtr@markwordseps		
..... 231–233		
\@glsxtr@mgl@or@gls@gcs		
..... 427		
\@glsxtr@mgl@or@gls@mcs		
..... 427		
\@glsxtr@mgl@label		
..... 425		
\@glsxtr@mgl@like		
..... 426		
\@glsxtr@mgl@likelist		
..... 426		
\@glsxtr@mgl@sreflist		
..... 425		

\@glxtr@org@glxtrtitlefirstplural	366, 368	\@glxtr@record	14–16, 78, 79, 85, 236, 238–246
\@glxtr@org@glxtrtitlefull ..	366, 368	\@glxtr@record@noglossarywarning ..	15
\@glxtr@org@glxtrtitlefullpl	366, 368	\@glxtr@record@only@setup	15, 16
\@glxtr@org@glxtrtitlelong ..	366, 368	\@glxtr@record@setting	
\@glxtr@org@glxtrtitlelongpl	366, 368	7, 9–11, 14, 15, 32, 63, 68, 69,
\@glxtr@org@glxtrtitlename ..	366, 367		106, 139, 157–159, 161, 162, 174, 425, 426
\@glxtr@org@glxtrtitleorpdforheading	366, 367	\@glxtr@record@setting@alsoindex ..	
\@glxtr@org@glxtrtitleplural	366, 367	7, 9, 15, 16, 63, 139
\@glxtr@org@glxtrtitleshort ..	366, 367	\@glxtr@record@setting@nameref ...	
\@glxtr@org@glxtrtitleshortpl	366, 367	10, 11, 14, 32, 159, 161
\@glxtr@org@glxtrtitletext ..	366, 367	\@glxtr@record@setting@off	
\@glxtr@org@makeglossaries ...	138, 139	68, 106, 174, 425, 426
\@glxtr@org@markboth	365	\@glxtr@record@setting@only	14, 32
\@glxtr@org@markright	365	\@glxtr@recordsee	14, 30, 63
\@glxtr@org@newacronymstyle ..	135, 137	\@glxtr@redef@forglsentries	6, 31
\@glxtr@org@postdescription	145, 223, 224	\@glxtr@redefstyles	24, 25, 437
\@glxtr@org@see@noindex	159	\@glxtr@reg@glosslist	139–142, 147
\@glxtr@org@setacronymlists	137	\@glxtr@restore@postpunc	144
\@glxtr@org@setacronymstyle ..	135, 137	\@glxtr@rglstrigger@record ...	179, 180
\@glxtr@org@theHvalue	7–9	\@glxtr@s@longnewglossaryentry	51
\@glxtr@org@unset@buffer	119, 120	\@glxtr@savepreloctag	75, 76
\@glxtr@org@prefix	10	\@glxtr@seefirstitem	61
\@glxtr@org@printglossary	72, 145	\@glxtr@seeitem	61
\@glxtr@org@warndep	231	\@glxtr@setentrycountunsetattr ...	121
\@glxtr@p@acrlong@	110	\@glxtr@setentryunitcountunsetattr	133
\@glxtr@p@acrlongpl@	110	\@glxtr@setupshortcuts	22, 23, 31
\@glxtr@p@acrshort@	110	\@glxtr@shortcutsnr	22
\@glxtr@p@acrshortpl@	110	\@glxtr@shortcutsval	22, 161
\@glxtr@p@long@	110	\@glxtr@showtargetsnr	27, 28
\@glxtr@p@longpl@	110	\@glxtr@showtargetsval	27
\@glxtr@p@plural@	110	\@glxtr@swaptwo	228
\@glxtr@p@short@	110	\@glxtr@tag	222
\@glxtr@p@shortpl@	110	\@glxtr@taggingcs	222
\@glxtr@p@text@	110	\@glxtr@textformat	84
\@glxtr@p@pagetag	75, 76	\@glxtr@theHentrycounter	10
\@glxtr@p@pagetag	75, 76	\@glxtr@theHvalue ..	7–9, 80, 83, 85, 177, 178
\@glxtr@p@prefix	449	\@glxtr@theentrycounter	10
\@glxtr@p@prevunitcount	130	\@glxtr@thevalue ..	7–9, 80, 83, 85, 177, 178
\@glxtr@p@printglossnr	145	\@glxtr@thisloctag	76
\@glxtr@p@printglossopts ..	72, 142, 145, 167	\@glxtr@titlelabel	149, 150, 171
\@glxtr@p@printglossval	145	\@glxtr@tmp	24, 25, 81, 152
\@glxtr@p@printunsrtglossaryskipentry	166, 169, 170	\@glxtr@type	207
\@glxtr@provide@addstoragekey	35	\@glxtr@unitcountlist	128
\@glxtr@provide@storagekey	34	\@glxtr@unset	119, 120
\@glxtr@providenewgls	174	\@glxtr@unset@buffer	119, 120
		\@glxtr@unsrt@getgrouptitle ..	166, 169
		\@glxtr@use@equation@counter ..	8, 81, 83
		\@glxtr@usesee	58

<code>\@glstrwarn@hybrid@noprintgloss</code>	139	<code>\@glstrsetaliasnoindex</code>	103, 104
<code>\@glstrwarn@onexistsordo</code>	6, 14, 16	<code>\@glstrshort</code>	110, 239, 430
<code>\@glstrwarn@undefaction</code>	6, 14, 16	<code>\@glstrshortpl</code>	110, 243
<code>\@glstr@wrglossary@locationhyperlink</code>	26	<code>\@glstrshowtargetleft</code>	27, 28
<code>\@glstr@wrglossnr</code>	80	<code>\@glstrshowtargetmark</code>	27, 28
<code>\@glstr@wrglossval</code>	80	<code>\@glstrshowtargetright</code>	28
<code>\@glstrbuffer@nodup@unset</code>	119	<code>\@glstrundeftag</code>	5, 33
<code>\@glstrbuffer@unset</code>	119	<code>\@glstrwrglossmark</code>	26, 27
<code>\@glstrdialecthook</code>	437	<code>\@gobble</code>	6, 15, 18, 86, 136, 140, 170, 171, 230, 405, 406, 410, 411, 418, 489, 529–532
<code>\@glstrdocdeffalse</code>	69	<code>\@gobblefour</code>	397
<code>\@glstrentryfmt</code>	37	<code>\@gobbletwo</code>	137, 231, 393
<code>\@glstrfmt</code>	35	<code>\@gtempa</code>	439
<code>\@glstrforcsvfield</code>	38	<code>\@ifdefinable</code>	396, 440
<code>\@glstrglossentry</code>	163	<code>\@ifglossaryexists</code>	33
<code>\@glstrglossentryother</code>	164	<code>\@ifnextchar</code>	105, 146, 427
<code>\@glstrhypernameprefix</code>	145, 146, 171	<code>\@ifpackageloaded</code>	4, 20, 161, 183, 208, 209, 212, 213, 215, 437, 455, 486
<code>\@glstrifhasfield</code>	39–41, 46, 47	<code>\@ifstar</code>	33–35, 38–42, 46, 47, 51, 52, 54, 69, 105, 119, 165, 222, 427, 439
<code>\@glstrifhyphenstart</code>	344	<code>\@ifundefined</code>	437, 439
<code>\@glstrindexaliased</code>	103	<code>\@ignored@glossaries</code>	53–55
<code>\@glstrindexcrossrefsfalse</code>	17	<code>\@input</code>	159
<code>\@glstrindexcrossrefstrue</code>	17	<code>\@input@</code>	154
<code>\@glstrinmark</code>	365	<code>\@istfilename</code>	139
<code>\@glstrlong</code>	110, 241, 430	<code>\@makeglossaries@warn@noprintglossary</code>	140
<code>\@glstrlongpl</code>	110, 245	<code>\@makeglossary</code>	139, 140
<code>\@glstrmultientryadjustedname</code>	451, 452	<code>\@mfu@domakefirstuc</code>	222, 223
<code>\@glstrmultientryadjustednamefirstfmt</code>	451, 452	<code>\@mfu@nocaplist</code>	223
<code>\@glstrmultientryadjustednamefirstother</code>	451, 452	<code>\@mgls@all</code>	407, 416, 417
<code>\@glstrmultientryadjustednamefmt</code>	451, 452	<code>\@mgls@attroptions</code>	409
<code>\@glstrmultientryadjustednameother</code>	451, 452	<code>\@mgls@combinedlink</code>	410
<code>\@glstrmultientryadjustednamepostsep</code>	451, 452	<code>\@mgls@cs</code>	417
<code>\@glstrmultientryadjustednamepresep</code>	451, 452	<code>\@mgls@current@iffirstuse</code>	415–418
<code>\@glstrmultientryadjustednamesep</code>	451–453	<code>\@mgls@current@options</code>	415–417
<code>\@glstrnewgls</code>	175, 176	<code>\@mgls@disable@writeseparateref@cond</code>	425
<code>\@glstrnewgls@inner</code>	174	<code>\@mgls@do@current@element</code>	405, 406, 410, 411, 415, 418
<code>\@glstrnewgls@innercsname</code>	174, 175	<code>\@mgls@dooptions</code>	407, 409, 410
<code>\@glstrnotinmark</code>	365	<code>\@mgls@hyper</code>	415–417
<code>\@glstrp</code>	115, 116	<code>\@mgls@hyper@nr</code>	401
<code>\@glstrp@opt</code>	113	<code>\@mgls@hyper@val</code>	401
<code>\@glstrpl</code>	71, 72	<code>\@mgls@hyperlink</code>	402, 410
<code>\@glstrpostloctag</code>	74, 75, 77	<code>\@mgls@main</code>	416
<code>\@glstrpreloctag</code>	74, 75, 77	<code>\@mgls@options</code>	409
		<code>\@mgls@others</code>	417

<code>\@mgls@post@hookdefs</code>	<code>\@nocounterr</code>	183
..... 408, 411, 412, 414, 417–419	<code>\@nopostdesc</code>	144, 168
<code>\@mgls@previous@iffirstuse</code>	<code>\@onelevel@sanitize</code> .	10, 12, 63, 72, 150, 171
417, 418	<code>\@onlypreamble</code>	73,
<code>\@mgls@previouslabel</code> 413, 417, 418, 452, 453	75, 132, 159, 162, 183, 216, 218, 219, 222	
<code>\@mgls@resetall</code>	<code>\@org@glossaryentrynumbers</code>	
404, 407 143, 144, 167, 168	
<code>\@mgls@resetall@nr</code>	<code>\@org@newglossaryentryprehook</code>	52
403	<code>\@pgls@</code>	434
<code>\@mgls@resetall@val</code>	<code>\@pglsorgls@</code>	435
403	<code>\@pglsorglspl@</code>	435
<code>\@mgls@resetmain</code>	<code>\@pglspl@</code>	434
404, 407	<code>\@print@unsrt@glossary</code>	165
<code>\@mgls@resetmain@nr</code>	<code>\@print@unsrt@innerglossary</code>	167
403	<code>\@printgloss@checkexists</code>	143, 165
<code>\@mgls@resetmain@val</code>	<code>\@printgloss@checkexists@allowignored</code>	
403 165	
<code>\@mgls@resetothers</code>	<code>\@printgloss@setsort</code>	142, 143, 168
405, 407	<code>\@printglossary</code>	72, 165
<code>\@mgls@resetothers@nr</code>	<code>\@printunsrt@glossary@handler</code> .	167, 169
403, 404	<code>\@printunsrtglossary</code>	165
<code>\@mgls@resetothers@val</code>	<code>\@provide@multi@glossaryentry@noop</code>	393
403	<code>\@rGLS</code>	180
<code>\@mgls@setup</code>	<code>\@rGLS@</code>	180
407, 409	<code>\@rGLSpl</code>	180
<code>\@mgls@setup@do</code>	<code>\@rGLSpl@</code>	180
401	<code>\@rGls</code>	179
<code>\@mgls@setup@do@not</code>	<code>\@rGls@</code>	179
401	<code>\@rGlspl</code>	179
<code>\@mgls@setupoptions</code>	<code>\@rGlspl@</code>	180
407, 409, 410	<code>\@rc@ifdefinable</code>	440
<code>\@mgls@thislabel</code>	<code>\@rgls</code>	178
398, 399	<code>\@rgls@</code>	178
<code>\@mgls@unsetaction</code>	<code>\@rglspl</code>	179
419	<code>\@rglspl@</code>	179
<code>\@mgls@unsetaction@val</code>	<code>\@sGlsXtrEnableOnTheFly</code>	69
401	<code>\@secondofthree</code>	87–89, 94, 96–98,
<code>\@mgls@unsetall</code>	100, 237, 238, 240, 242, 244, 246, 418, 419	
403, 407	<code>\@secondoftwo</code>	27, 79, 86, 89–98,
<code>\@mgls@unsetall@nr</code>	101, 109, 137, 145, 172, 214, 228, 236,	
404	237, 239–246, 262, 264, 287, 289, 304,	
<code>\@mgls@unsetall@val</code>	306, 328, 330, 366, 367, 397, 402, 406,	
404	408, 409, 411–413, 415, 418, 419, 489, 490	
<code>\@mgls@unsetmain</code>	<code>\@sglsxtr@provide@storagekey</code>	34
403, 407	<code>\@star@or@long</code>	439
<code>\@mgls@unsetmain@nr</code>	<code>\@starttoc</code>	365
404	<code>\@thirdofthree</code>	87–89, 95–98, 101,
<code>\@mgls@unsetmain@val</code>	237, 239, 241, 242, 244, 246, 367, 418, 419	
404	<code>\@thirdoftwo</code>	90–94
<code>\@mgls@unsetothers</code>		
404, 407		
<code>\@mgls@unsetothers@nr</code>		
404		
<code>\@mgls@unsetothers@val</code>		
404		
<code>\@mgls@writeseparatorrefsfalse</code>		
425		
<code>\@mgls@writeseparatorrefstrue</code>		
425		
<code>\@mglslocalreset</code>		
398		
<code>\@mglslocalunset</code>		
398		
<code>\@mglsreset</code>		
397		
<code>\@mglsunset</code>		
397		
<code>\@multi@glossary@doifexists</code> ...		
394, 397		
<code>\@multi@glossary@entry</code>		
392, 393, 397		
<code>\@multi@glossaryentry</code>		
392, 393		
<code>\@multi@glossaryentry@list</code> .		
396, 398, 399		
<code>\@multi@glossentry@donext</code>		
392, 393		
<code>\@multiglossaryentry</code>		
392		
<code>\@ne</code>		
124, 133, 174		
<code>\@newglossaryentry@defcounters</code>		
122, 130		
<code>\@newglossaryentryposthook</code> .		
13, 34, 64, 99		
<code>\@newglossaryentryprehook</code>		
13, 34, 52, 63, 99		
<code>\@nil</code>		
153, 172, 444		
<code>\@nnil</code>		
61, 217, 218, 220, 228–230, 413		
<code>\@no@glxtrindexaliased</code>		
103, 104		
<code>\@no@makeglossaries</code>		
158		

\@this@key	214	320, 322, 323, 326, 329, 332, 334, 336,
\@tracklang@lang	48, 49	339, 342, 345, 347, 351, 354, 357, 360, 362
\@warn@nomakeglossaries	154	\ABP 20
\@xGlsXtrIfValueInFieldCsvList	40	\Abp 20
\@xdy@main@language	154	\abp 20
\@xdycrossrefhook	62	\AC 21
\@xdy@language	154	\Ac 21
\@xdylocationclassorder	62	\ac 21
\@xfor@nextelement	61, 413	\ACF 2
[package	442	\Acf 21
\\	153, 444	\acf 21
		\ACFP 21
		\Acfp 21
		\acfp 21
		\ACL 21
		\Acl 21
		\acl 21
		\ACLP 21
		\Aclp 21
		\aclp 21
		\ACP 21
		\Acp 21
		\acp 21
		\ACRfull 99
		\Acrfull 99
		\acrfull 98
		\ACRfullfmt 99, 135
		\Acrfullfmt 99, 135
		\acrfullfmt 99, 135
		\ACRfullpl 99
		\Acrfullpl 99
		\acrfullpl 99
		\ACRfullplfmt 99, 135
		\Acrfullplfmt 99, 135
		\acrfullplfmt 99, 135
		\ACRlong 97
		\Acrlong 97
		\acrlong 96
		\ACRlongpl 98
		\Acrlongpl 98
		\acrlongpl 97
		\acronymentry 134
		\acronymfont 94–98, 112, 113, 136, 137
		\acronymname 20
		\acronymsort 134
		\acronymtype 20, 134, 136, 438
		\acrpluralsuffix 134, 161
		\ACRshort 95
		\Acrshort 94
A		
\AA	469	
\aa	469	
\AB	20	
\Ab	20	
\ab	20	
abbreviation styles:		
footnote	261	
long-hyphen-postshort-hyphen	349, 350, 352	
long-hyphen-short-hyphen	346, 350	
long-postshort-sc-user	338	
long-postshort-user	337	
long-short	192	
long-short-sc-user	335	
long-short-user	333	
nolong-short	269	
short	266	
short-em-footnote	327	
short-em-postfootnote	330	
short-footnote	260, 302	
short-hyphen-long-hyphen	354, 356	
short-hyphen-postlong-hyphen	355, 356, 358	
short-long-user	338	
short-nolong	266, 268	
short-nolong-desc	268	
short-postfootnote	263	
short-postlong-user	340	
short-sc-footnote	285, 288	
short-sm-postfootnote	305	
\abbreviationsname	20	
\abbrvpluralsuffix		
. 161, 232, 254, 256, 259, 262, 264, 267,		
270, 273, 275, 277, 279, 281, 282, 284,		
287, 290, 292, 294, 295, 298, 299, 301,		
304, 307, 309, 311, 313, 314, 316, 318,		

<code>\cGls</code>	20, 21, 121, 133	280, 284, 286, 288, 290–295, 297, 301–
<code>\cglS</code>	20, 21, 121, 133	303, 305, 306, 308, 310–314, 316, 318,
<code>\cGLSformat</code>	125	320, 323, 325, 327, 328, 330, 332, 333,
<code>\cGlsformat</code>	125	335, 337, 338, 340–343, 345–347, 349,
<code>\cglSformat</code>	124, 126	350, 352, 354–356, 358, 359, 361, 362, 364
<code>\cGLSpl</code>	20, 21, 121, 133	
<code>\cGlspl</code>	20, 21, 121, 133	D
<code>\cglSpl</code>	20, 21, 121, 133	datatool-base package
<code>\cGLSplformat</code>	126	11
<code>\cGlsplformat</code>	125	<code>\DeclareAcronymList</code>
<code>\cglSplformat</code>	125, 127	134
<code>\char</code>	149	<code>\DeclareOption</code>
<code>\columnwidth</code>	74	4, 487
<code>\count@</code>	123, 124, 132, 133, 399	<code>\DeclareOptionX</code>
<code>\csappto</code>	44, 50, 400	4, 28
<code>\csdef</code>	34, 44, 45,	<code>\def</code>
	50, 99–101, 123, 128, 129, 131, 196, 201,	8, 10, 12–17, 27–29, 32,
	213, 224, 225, 250–252, 261, 263, 287,	33, 36, 38, 43, 52, 54, 58, 63, 66, 70–72,
	288, 303, 305, 328, 330, 333, 335, 337–	74, 77–83, 85, 87–99, 106, 109–113, 119,
	340, 351, 352, 356, 358, 400, 420, 488, 509	124–127, 134, 140, 142, 143, 145–153,
<code>\csedef</code>	129, 394, 395, 399, 400, 560	166, 167, 169, 171, 172, 174, 177–180,
<code>\csgdef</code>	45, 53–55,	191–194, 217–220, 222, 223, 227–232,
	68, 75, 123, 129, 131, 132, 136, 201, 509, 532	236–246, 250, 344, 347, 349, 350, 353,
<code>\cslet</code>	45, 52, 143, 394–396, 399, 400	355, 356, 392, 393, 397, 409–411, 413,
<code>\csletcs</code>	45, 252, 395	414, 417, 426, 427, 430, 432–434, 442–
<code>\csname</code>	5, 35,	444, 448, 449, 452, 486, 501, 518–520,
	54, 63, 68, 73, 77, 79, 82, 85, 94–98, 100,	524–526, 529, 530, 536, 537, 557, 561, 562
	101, 103, 113, 129, 140, 150, 154, 157,	<code>\defglSentryfmt</code>
	158, 166–169, 174–178, 182, 207, 231,	53–55
	236–246, 252, 396–398, 402, 426, 490, 518	<code>\define@boolkey</code>
<code>\cspreto</code>	50	17, 18, 80, 102, 146, 390, 391, 401
<code>\csundef</code>	202	<code>\define@choicekey</code>
<code>\csuse</code>	8,	6, 15, 17, 19, 22,
	11, 36, 37, 44, 51, 54, 65, 75, 100, 101,	24, 27, 80, 145, 388, 389, 391, 401, 403, 404
	115–117, 128–130, 132, 143, 149, 151,	<code>\define@key</code> ..
	152, 163, 164, 171, 172, 202, 213, 224,	13, 19, 24, 25, 34, 63, 80, 81,
	225, 250, 252, 391, 396, 410, 420, 421,	85, 99, 145, 146, 191, 229, 389, 390, 400, 401
	442, 444, 447, 509, 510, 512, 513, 536, 560	<code>\DefineAcronymSynonyms</code>
<code>\csxdef</code> ...	57, 64, 129, 132, 394, 395, 399, 400	22
<code>\currentglossary</code>		<code>\delimN</code>
	143, 163, 164, 167, 168, 529, 532	75, 76
<code>\CurrentOption</code>	28, 487, 488	<code>\delimR</code>
<code>\CurrentTrackedLanguage</code>	485	75, 76
<code>\CurrentTrackedLanguageTag</code>	161	<code>\descriptionname</code>
<code>\CurrentTrackedScript</code>	485, 486	536, 539, 541,
<code>\CurrentTrackedTag</code>	437, 486	542, 544, 545, 547, 548, 550, 552, 553, 555
<code>\CustomAbbreviationFields</code>		<code>\detokenize</code>
	233, 253, 255, 257, 258,	70
	260, 261, 263, 264, 266, 269, 272–278,	<code>\dimen@</code>
		138, 423, 424, 509–517, 537
		<code>\dimen@i</code>
		512, 513
		<code>\dimen@ii</code>
		509, 510, 512, 513, 537
		<code>\dimexpr</code>
		73, 74, 508, 537, 538, 559
		<code>\disable@keys</code>
		20, 31, 33, 69, 159
		<code>\do</code>
		5, 24, 39, 61,
		66, 86, 121, 122, 133, 136, 137, 139, 142,
		150, 166, 169, 177, 183, 201, 207, 214,
		222, 394, 398, 399, 402–405, 413, 449, 452
		<code>\do@glS@link@checkfirsthyper</code>
		36, 78, 79, 83, 94–98, 236–246
		<code>\do@glSdisablehyperinlist</code>
		83, 103
		doc package
		219
		<code>\dolistcsloop</code>
		38

<code>\DTLifinlist</code>	39–41, 82, 107, 136, 140, 141, 147, 425, 448
<code>\DTLifint</code>	149
E	
<code>\eappto</code> 24, 194, 195, 396, 407, 409, 415–417, 487	
<code>\edef</code>	10, 48, 67, 149, 153, 154, 159, 182, 217, 218, 220, 407, 409, 410, 415, 416, 426, 444, 449, 485, 529–531, 538–552, 554, 555, 562
<code>\eglssetwidest</code>	511–517
<code>\egroup</code>	52, 144, 397, 419, 451, 452
<code>\else</code>	7–12, 14, 17–20, 22–24, 29, 30, 32, 36, 43, 49, 50, 61, 67–70, 75, 77, 84, 102–104, 124, 136, 138, 139, 141, 143–145, 149, 151–153, 155–157, 160, 168, 172, 177, 178, 216–218, 220, 228, 230, 232, 239–246, 249, 254, 256, 257, 259, 260, 262, 263, 265, 267–271, 274, 276–285, 287, 288, 290–302, 304, 305, 307, 309, 311, 313, 315–327, 329, 332–334, 336, 339, 340, 342–344, 347, 350, 352, 353, 356–358, 360, 362, 363, 392–396, 399, 400, 402–407, 409, 410, 413, 414, 416–418, 423–426, 444–447, 452, 490, 493–500, 502, 505, 506, 518, 519, 526, 528–531, 539–556, 561
<code>\emph</code>	192, 306
<code>\empty</code> 28, 151–153, 192, 409, 417, 444, 448, 452	
<code>\encapchar</code>	220
<code>\end</code>	151, 156, 167, 168, 490, 492–499, 521, 522, 524, 525, 530, 539–546, 548, 549, 551–556, 562
<code>\end@getprefix</code>	32
<code>\end@glxtr@display@loc</code>	151
<code>\endcsname</code>	5, 35, 54, 63, 68, 73, 77, 79, 82, 85, 94–98, 100, 101, 103, 113, 129, 140, 150, 154, 157, 158, 166–169, 174–178, 182, 207, 231, 236–246, 252, 396–398, 402, 426, 490, 518
<code>\endfoot</code>	536, 539, 541, 542, 544, 545, 547, 548, 550, 551, 553, 555
<code>\endgroup</code> 7, 9, 36, 38, 82, 86, 103, 163, 165, 167, 182, 193, 221, 439, 443, 444, 449, 558	
<code>\endhead</code>	536, 539, 541, 542, 544, 545, 547, 548, 550, 551, 553, 555
<code>\ensuremath</code>	26
entry categories:	
abbreviation	231, 246
general	200, 203
index	205
nameshortaccess	316
<code>\entryname</code>	536, 537, 539, 541, 542, 544, 545, 547, 548, 550, 552, 553, 555
<code>\epreto</code>	217
<code>\equal</code>	157, 158, 225
<code>equation (counter)</code>	18, 81
<code>\escapechar</code>	439
<code>etoolbox</code> package	4
<code>\everypar</code>	559
<code>\expandafter</code>	28, 35, 36, 39–41, 48, 58, 60, 61, 66, 70–72, 81, 100, 101, 105, 107, 113, 119, 126–128, 136, 140–142, 147, 150, 152, 153, 166, 168, 169, 172–175, 181, 193, 194, 207, 210, 211, 214–216, 219, 220, 228, 231–233, 344, 396–398, 402, 417, 425, 427, 439, 444, 490
<code>\expandonce</code> ..	134, 172, 193, 194, 217, 218, 254, 347, 393, 396, 407, 409–411, 414, 417, 419, 425, 426, 490, 538–552, 554–556
<code>\ExtraCustomAbbreviationFields</code>	194, 195, 231, 233
F	
<code>\fi</code>	6–12, 14, 16–20, 22–24, 27–32, 36, 43, 50, 57, 61, 63–65, 67, 69, 70, 73–75, 77, 80, 83, 84, 102–104, 124, 132, 133, 136, 138, 139, 142–146, 149, 151–162, 166–169, 172, 177, 178, 216–218, 220, 228, 230, 233, 239–246, 249, 254, 256, 257, 259, 260, 262, 263, 265, 267–271, 274–285, 287, 288, 290–302, 304, 305, 307, 309, 311, 313–327, 329, 332–334, 336, 339, 340, 342–344, 347, 350, 352, 353, 356–358, 360, 362, 363, 392–397, 399, 400, 402–407, 410–419, 423–426, 438, 444–447, 452, 490, 493–500, 502, 504–506, 509–519, 526, 528–531, 537, 539–551, 553, 554, 556, 557, 561, 562
<code>\firstacronymfont</code>	136–138
<code>fontspec</code> package	161
<code>\footnote</code>	258
<code>\footnotesize</code>	29
<code>\forall</code> abbreviation lists	136
<code>\forall</code> glossaries 66, 165, 205, 207, 510–517	
<code>\forall</code> glossaries	67, 123, 132
<code>\ForEachTrackedDialect</code>	437, 486
<code>\foreignlanguage</code>	47, 49
<code>\for</code> glossaries	5, 66, 205, 207, 510–517
<code>\forlistcsloop</code>	38, 132, 151

<code>\forlistloop</code>	120, 148, 223	<code>index</code>	501
<code>\futurelet</code>	228	<code>indexgroup</code>	502
G			
<code>\gappto</code>	411, 412, 418, 419	<code>indexhypergroup</code>	502
<code>\gdef</code>	76, 107, 218, 219, 393, 408, 425, 426	<code>inline</code>	500
<code>\Genacrfullformat</code>	135	<code>list</code>	490
<code>\genacrfullformat</code>	135	<code>listdotted</code>	488
<code>\GenericAcronymFields</code>	134	<code>listdottedstyle</code>	489
<code>\Genplacrfullformat</code>	135	<code>listgroup</code>	491
<code>\genplacrfullformat</code>	135	<code>listhypergroup</code>	491
<code>\GetTitleStringDisableCommands</code>	489	<code>longragged-booktabs</code>	538
<code>\GetTitleStringSetup</code>	489	<code>mcolalttree</code>	524
<code>\GetTrackedDialectFromLanguageTag</code> 47, 48		<code>mcolalttreegroup</code>	524
<code>\GetTrackedDialectToMapping</code>	48	<code>mcolalttreehypergroup</code>	525
<code>\glo@grabfirst</code>	172	<code>mcolalttreespannav</code>	525
<code>\glo@name</code>	210, 211, 215	<code>mcolindexgroup</code>	520
<code>\gloaliaslabel</code>	108	<code>mcolindexhypergroup</code>	521
<code>\global</code> . 10, 52, 67, 144, 168, 172, 394–399, 425		<code>mcolindexspannav</code>	521
<code>\glolinkprefix</code>		<code>mcoltreegroup</code>	521
..... 81, 82, 84, 108, 109, 145, 161, 402		<code>mcoltreehypergroup</code>	522
<code>glossaries package</code>	11, 15, 29,	<code>mcoltreenonamegroup</code>	523
30, 32, 49, 51, 57, 62, 63, 65, 102, 106,		<code>mcoltreenonamehypergroup</code>	523
138, 139, 142, 143, 165, 192, 207, 488, 490		<code>mcoltreenonamespannav</code>	523
<code>glossaries-accsupp package</code> 23, 28, 183, 192, 233		<code>mcoltreespannav</code>	522
<code>glossaries-extra package</code>	2, 51, 486	<code>name-desc</code>	541
<code>glossaries-extra-bib2gls package</code>		<code>sublistdotted</code>	489
..... 15, 32, 437, 486, 537		<code>tree</code>	503
<code>glossaries-extra-stylemods package</code>		<code>treegroup</code>	505
..... 24, 224, 437, 445		<code>treehypergroup</code>	505
<code>glossaries-prefix package</code>	23, 28, 434, 449	<code>treenoname</code>	505
<code>glossaries-stylemods package</code>	527	<code>treenonamegroup</code>	507
<code>glossaries.sty package</code>	52	<code>treenonamehypergroup</code>	507
<code>\GlossariesExtraWarning</code>	5,	<code>glossary-bookindex package</code>	488
16, 18, 24, 32, 48–50, 70, 72, 84, 94,		<code>glossary-hypernav package</code>	106
137, 141, 152, 156, 158, 159, 166, 169,		<code>glossary-list package</code>	489, 490
208–214, 221–223, 252, 407, 434, 439, 447		<code>glossary-long package</code>	494
<code>\GlossariesExtraWarningNoLine</code>		<code>glossary-longbooktabs package</code>	494
..... 14, 18, 124, 133, 162		<code>glossary-longextra package</code>	445–447
<code>\GlossariesWarning</code>		<code>glossary-tree package</code>	446, 447, 501, 560
..... 75, 134, 141, 143, 148, 149, 167, 250		<code>\glossaryentrynumbers</code>	
<code>\GlossariesWarningNoLine</code> ... 107, 138, 154	 77, 143, 144, 167, 168, 173	
<code>glossary styles:</code>		<code>\glossaryheader</code>	
<code>altlist</code>	490 151, 166, 168, 490–499, 501,	
<code>altlistgroup</code>	491	502, 504–507, 518, 520–523, 525, 530,	
<code>altlisthypergroup</code>	492	539, 540, 542–546, 548, 549, 551–557, 561	
<code>alttree</code> 445, 446, 503, 507, 508, 518, 536		<code>\glossaryname</code>	143
<code>alttreegroup</code>	519	<code>\glossarypostamble</code> 151, 167, 168, 171	
<code>alttreehypergroup</code>	519	<code>\glossarypreamble</code>	151, 166, 168
		<code>\glossarysection</code> 151, 157, 158, 166, 168, 171	
		<code>\glossarytitle</code> 54, 143, 151, 157, 158, 166–168	

<code>\glossarytoctitle</code>	309, 311, 313, 314, 316, 318, 320, 322, 323, 326, 329, 332, 334, 336, 339, 341– 343, 345, 347, 350, 351, 354, 357, 360, 362
<code>\glossentry</code>	144, 168, 173, 488, 490, 492–499, 501, 504, 506, 518, 530, 539, 540, 542, 543, 545, 546, 548, 549, 551, 553, 554, 556, 557, 561
<code>\Glossentrydesc</code>	559
<code>\glossentrydesc</code> 488, 489, 492–500, 503–506, 534, 560
<code>\Glossentryname</code>	558
<code>\glossentryname</code>	163, 488, 489, 492– 499, 502, 504, 506, 518, 519, 527, 534, 560
<code>\glossentrynameother</code>	165
<code>\glossentrysymbol</code>	494, 496–500, 504, 506, 508, 534, 558, 560
<code>\glossxtrsetpopts</code>	224
<code>\GLS</code>	121, 133, 177
<code>\Gls</code>	71, 121, 133, 176
<code>\gls</code>	49, 50, 71, 72, 121, 133, 141, 155, 176
<code>\gls@assign@desc</code>	52
<code>\gls@assign@field</code>	13, 34, 99
<code>\gls@checkseeallowed</code>	67, 69, 140
<code>\gls@codepage</code>	154
<code>\gls@defdocnewglossaryentry</code> .	67, 122, 130
<code>\gls@defglossaryentry</code>	52, 68, 71, 72
<code>\gls@dotocitle</code>	143, 167, 168
<code>\gls@glossary</code>	63
<code>\gls@grplabel</code>	106
<code>\gls@hypergroup prerun</code>	107
<code>\gls@ifnotmeasuring</code>	11, 121, 397, 398
<code>\gls@level</code>	172, 173
<code>\gls@noidxglossary</code>	140
<code>\gls@org@glossaryentryfield</code> ...	144, 168
<code>\gls@org@glossarysubentryfield</code>	144, 168
<code>\gls@orgTrackLangRequireDialectPrefix</code>	485, 486
<code>\gls@save@numberlist</code>	74, 75, 77
<code>\gls@set@xr@key</code>	63
<code>\gls@tmplen</code>	510–517, 519, 537, 538
<code>\gls@type</code>	140
<code>\glsabbrvdefaultfont</code> ...	235, 254, 256, 259, 262, 264, 267, 270, 331, 344, 347, 359
<code>\glsabbrvmfont</code>	306–314, 316, 318, 320, 322, 323, 325–330
<code>\glsabbrvfont</code> 111, 112, 136, 239–241, 243, 244, 247, 249, 253–264, 266, 267, 270, 271, 273, 275, 277, 279, 281, 282, 284, 287, 290, 292, 294, 295, 297, 299, 301, 304, 307,
<code>\glsabbrvhyphenfont</code>	309, 311, 313, 314, 316, 318, 320, 322, 323, 326, 329, 332, 334, 336, 339, 341– 343, 345, 347, 350, 351, 354, 357, 360, 362
<code>\glsabbrvonlyfont</code>	345, 346, 350–352, 354–358
<code>\glsabbrvscfont</code>	359–361
<code>\glsabbrvscfont</code>	273–279, 281, 282, 284, 286–288, 334, 361
<code>\glsabbrvsconlyfont</code>	361, 362, 364
<code>\glsabbrvscuserfont</code>	335–338
<code>\glsabbrvsmfont</code>	289–295, 297, 299, 301, 303–305
<code>\glsabbrvuserfont</code>	331–334, 337, 339, 340, 342
<code>\GLSaccessdesc</code>	90
<code>\Glsaccessdesc</code>	90, 208, 220
<code>\glsaccessdesc</code>	90, 208, 226
<code>\GLSaccessdescplural</code>	91
<code>\Glsaccessdescplural</code>	90
<code>\glsaccessdescplural</code>	90
<code>\GLSaccessfirst</code>	88
<code>\Glsaccessfirst</code>	88
<code>\glsaccessfirst</code>	87
<code>\GLSaccessfirstplural</code>	89
<code>\Glsaccessfirstplural</code>	89
<code>\glsaccessfirstplural</code>	89
<code>\Glsaccesslong</code>	97, 234, 242, 254, 265, 270, 274, 280, 281, 283, 291, 297–300, 307, 309, 317–322, 324, 333, 334, 336, 346, 348, 352, 354, 360, 363
<code>\glsaccesslong</code> 96, 97, 234, 241, 242, 254, 256, 259, 260, 262, 263, 265, 267, 268, 270, 271, 274, 276–283, 285, 288, 290, 292–294, 296–302, 304, 305, 307, 309, 311, 313, 315–327, 329, 332, 334, 336, 340, 342, 343, 346, 348, 351, 352, 354, 360, 362, 363
<code>\Glsaccesslongpl</code> 98, 234, 246, 254, 265, 270, 274, 280– 283, 291, 297–300, 307, 309, 317–324, 333, 334, 336, 346, 348, 352, 354, 360, 363
<code>\glsaccesslongpl</code> 97, 98, 234, 245, 246, 254, 256, 257, 259, 260, 262, 263, 265, 267, 268, 270, 271, 274, 276–283, 285, 288, 290, 292–294, 296–302, 305, 307, 309, 311, 313, 315–327, 329, 332, 334, 336, 340, 343, 346, 348, 351, 352, 354, 360, 362, 363
<code>\GLSaccessname</code>	90
<code>\Glsaccessname</code>	90

<code>\glsaccessname</code>	89	<code>\glsbackslash</code>	70
<code>\GLSaccessplural</code>	88	<code>\glscapitalisewords</code>	207, 453
<code>\Glsaccessplural</code>	88	<code>\glsapscase</code>	
<code>\glsaccessplural</code>	88	... 79, 86–98, 100, 101, 236–248, 418–420	
<code>\Glsaccessshort</code>	94, 240,	<code>\glscategory</code>	77, 86,
249, 256, 259, 260, 262, 263, 267, 268,		101, 111, 112, 202–204, 208, 209, 211–	
276, 278, 279, 285, 287, 288, 292, 294,		214, 220, 221, 224, 225, 236–241, 243, 244	
296, 302, 304, 305, 311, 313, 315, 316,		<code>\glscategorylabel</code>	
326, 327, 329, 339, 340, 343, 351, 357, 358		. 101, 102, 192, 194, 195, 229, 231–233,	
<code>\glsaccessshort</code>	94, 95, 234,	250, 253, 255, 257, 258, 260, 261, 263,	
239, 241, 249, 254, 256, 259, 262, 265,		264, 266, 271, 273–278, 284, 286–288,	
267, 268, 270, 274, 275, 277–281, 283–		290–293, 295, 297, 301–303, 305, 306,	
285, 287, 288, 290–292, 294–298, 300–		308–314, 316, 318, 319, 321, 325, 327,	
302, 304, 307, 309, 311, 313–317, 319–		328, 330, 332, 333, 335, 337–343, 345,	
322, 324, 326, 329, 332–334, 336, 339,		346, 349–353, 355, 356, 358, 359, 361–363	
340, 342, 346, 348, 351, 354, 357, 360, 363		<code>\glsclsebrace</code>	62, 156–158
<code>\Glsaccessshortpl</code>	96, 244, 249,	<code>\glscombinedfirstsep</code>	417, 422, 424
256, 259, 260, 262, 263, 267, 269, 276,		<code>\glscombinedfirstsepfir</code>	
278, 279, 285, 288, 293, 294, 296, 302,	 417, 422–424, 453	
304, 305, 311, 313, 315, 316, 326, 327,		<code>\glscombinedsep</code>	417, 421–423
329, 339, 340, 343, 351, 357, 358, 361, 363		<code>\glscombinedsepfir</code>	417, 422, 423
<code>\glsaccessshortpl</code>	95, 96,	<code>\glscounter</code>	8, 18
234, 243, 244, 249, 254, 256, 259, 262,		<code>\glscurrententrylabel</code>	74–
265, 267, 268, 270, 274, 276–283, 285,		76, 144, 153, 163, 164, 166–170, 223, 224	
287, 288, 290–292, 294–298, 300–302,		<code>\glscurrentfieldvalue</code>	36,
304, 307, 309, 311, 313, 315–317, 319–		37, 39–41, 43, 46–48, 59, 60, 331, 433, 440	
324, 326, 327, 329, 332–334, 336, 339,		<code>\glscustomtext</code> ..	78, 79, 94–98, 236–247, 249
340, 343, 346, 348, 351, 354, 357, 360, 363		<code>\glsdefaultshortaccess</code>	193, 194
<code>\GLSAccesssymbol</code>	91	<code>\glsdefaulttype</code>	
<code>\GlsAccesssymbol</code>	91, 221 5, 19, 50, 142, 143, 155, 165, 167, 170	
<code>\glsAccesssymbol</code>	91, 221, 226	<code>\glsdescriptionaccessdisplay</code> ..	187, 188, 208
<code>\GLSAccesssymbolplural</code>	92	<code>\glsdescriptionpluralaccessdisplay</code> ..	188
<code>\GlsAccesssymbolplural</code>	91	<code>\glsdescwidth</code>	492–499, 536–538
<code>\glsAccesssymbolplural</code>	91	<code>\glsdetoklabel</code>	7, 8, 33, 37, 38,
<code>\GLSAccessstext</code>	87	41, 43–45, 49–52, 56, 58, 60, 61, 66, 68,	
<code>\GlsAccessstext</code>	87	70, 82, 85, 103, 108, 122, 123, 128–132,	
<code>\glsAccessstext</code>	87	140, 144, 147, 148, 163, 164, 168, 172,	
<code>\glsacrshortcutstrue</code>	22, 23	176, 177, 207, 210, 211, 215, 490, 512, 513	
<code>\glsacspacemax</code>	138	<code>\glsdisp</code>	450
<code>\glsadd</code>	36, 66, 82, 86, 155	<code>\glsdisplaynumberlist</code>	141, 147
<code>\glsadd options</code>		<code>\glsdohyperlink</code>	106, 444
<code>theHvalue</code>	9	<code>\glsdohypertarget</code>	109, 145, 147
<code>thevalue</code>	8, 9	<code>\glsdoifexists</code>	17, 30, 44, 45, 51, 55,
<code>\glsaddpostsetkeys</code>	9, 85	58–61, 63, 78, 79, 85, 94–98, 109, 121,	
<code>\glsaddpresetkeys</code>	9, 85	140, 148, 149, 163, 164, 236–246, 393, 438	
<code>\glsaddstoragekey</code>	65, 200, 441	<code>\glsdoifexistsordo</code>	36, 37, 79
<code>\glsaltlistitem</code>	490	<code>\glsdoifexistsorwarn</code>	17, 208, 220, 221
<code>\glsalttreechildpredesc</code>	508	<code>\glsdoifnoexists</code>	52
<code>\glsalttreepredesc</code>	507, 508	<code>\glsdonohyperlink</code>	84, 109, 402

<code>\glsdosanitizesort</code>	141	<code>\Glsentrysymbol</code>	187, 198
<code>\glsdoshowtarget</code>	109	<code>\glsentrysymbol</code> .	186, 187, 198, 221, 514–516
<code>\glsenableentrycount</code>	121, 124, 132	<code>\Glsentrysymbolplural</code>	187, 198
<code>\glsenableentryunitcount</code>	123, 133	<code>\glsentrysymbolplural</code>	187, 198
<code>\glsentrycounter</code>	26, 152, 153, 444	<code>\Glsentrytext</code>	184, 197
<code>\GlsEntryCounterLabelPrefix</code>	49	<code>\glsentrytext</code> 109, 184, 197, 380, 381, 423, 424	
<code>\glsentrycurrcount</code>	122, 124, 130	<code>\glsentrytitlecase</code>	453
<code>\Glsentrydesc</code>	188, 199, 209	<code>\glsentrytype</code>	164, 233
<code>\glsentrydesc</code>	187, 188, 198, 199, 209	<code>\Glsentryuseri</code>	92
<code>\Glsentrydescplural</code>	188, 199	<code>\glsentryuseri</code>	92
<code>\glsentrydescplural</code>	188, 199	<code>\Glsentryuserii</code>	92
<code>\Glsentryfirst</code>	127, 181, 185, 198	<code>\glsentryuserii</code>	92
<code>\glsentryfirst</code>		<code>\Glsentryuseriii</code>	92
	127, 181, 185, 186, 198, 382, 383, 423, 424	<code>\glsentryuseriii</code>	93
<code>\Glsentryfirstplural</code> ...	127, 181, 186, 198	<code>\Glsentryuseriv</code>	93
<code>\glsentryfirstplural</code>		<code>\glsentryuseriv</code>	93
	127, 181, 186, 198, 383, 384	<code>\Glsentryuserv</code>	93
<code>\glsentryfmt</code>	53–55, 136	<code>\glsentryuserv</code>	93
<code>\Glsentryfull</code>	135	<code>\Glsentryuservi</code>	93
<code>\glsentryfull</code>	135	<code>\glsentryuservi</code>	94
<code>\Glsentryfullpl</code>	135	<code>\glsextrapostnamehook</code>	213
<code>\glsentryfullpl</code>	135	<code>\glsfieldfetch</code>	108
<code>\glsentryitem</code> ...	163, 164, 488, 489, 492–499, 501, 504, 506, 518, 530, 534, 558, 560	<code>\glsfieldxdef</code>	207
<code>\Glsentrylong</code> ...	112, 113, 127, 181, 190, 199	<code>\glsFindWidestLevelTwo</code>	447
<code>\glsentrylong</code> ...	112, 113, 127, 181, 190, 199, 200, 261, 263, 287, 289, 303, 305, 328, 330, 339, 341, 356, 384, 386, 423, 424	<code>\glsFindWidestTopLevelName</code>	447
<code>\Glsentrylongpl</code>	112, 113, 127, 190, 200	<code>\glsfindwidesttoplevelname</code>	510
<code>\glsentrylongpl</code>		<code>\GLSfirst</code>	373
	112, 113, 127, 190, 200, 385, 386	<code>\Glsfirst</code>	373
<code>\Glsentrylongplural</code>	181	<code>\glsfirst</code>	373
<code>\glsentrylongplural</code>	181	<code>\glsfirstabbrvdefaultfont</code>	
<code>\Glsentryname</code>	184, 197, 210–213, 453		235, 254, 256, 259, 262, 264, 267, 270, 347
<code>\glsentryname</code>	163, 183, 184, 197, 217, 380, 453, 511–517, 533	<code>\glsfirstabbrvmfont</code>	306–330
<code>\glsentrynumberlist</code>	141, 149, 515–517	<code>\glsfirstabbrvfont</code> ..	136, 234, 253–270, 273, 275, 277, 279, 281, 282, 284, 287, 290, 292, 294, 295, 298, 299, 301, 304, 307, 309, 311, 313, 314, 316, 318, 320, 322, 323, 326, 329, 332, 334, 336, 339, 342, 345, 347, 348, 351, 354, 357, 360, 362
<code>\glsentrypdfsymbol</code>	220	<code>\glsfirstabbrvhyphenfont</code>	
<code>\Glsentryplural</code>	185, 197		344–346, 350, 351, 353–358
<code>\glsentryplural</code>	185, 197, 381, 382	<code>\glsfirstabbrvonlyfont</code>	360, 361
<code>\glsentryprevcount</code>	122, 124, 130	<code>\glsfirstabbrvscfont</code>	273–288
<code>\glsentryprevmaxcount</code>	130	<code>\glsfirstabbrvsconlyfont</code>	362, 363
<code>\glsentryprevtotalcount</code>	130	<code>\glsfirstabbrvscuserfont</code> ...	335, 336, 338
<code>\Glsentryshort</code>	111, 112, 189, 199	<code>\glsfirstabbrvsmfont</code>	290–305
<code>\glsentryshort</code> 111, 112, 138, 189, 199, 333, 335, 337, 338, 350, 378, 379, 386, 423, 424		<code>\glsfirstabbrvuserfont</code>	
<code>\Glsentryshortpl</code>	112, 189, 199		332–334, 337, 339–343
<code>\glsentryshortpl</code>		<code>\glsfirstaccessdisplay</code>	185, 186
	111–113, 189, 190, 199, 379, 386		

<code>\glsfirstlongdefaultfont</code>	337–339, 341–343, 345–347, 349, 351, 353–356, 358, 359, 361, 362, 364, 421–424
254, 256, 265, 267, 270, 273–283, 290– 300, 306–308, 310–312, 314–319, 322, 323	
<code>\glsfirstlongemfont</code>	203, 409
.... 308–310, 312–314, 320, 321, 323–325	
<code>\glsfirstlongfont</code>	109, 440
..... 234, 253–257, 259, 262, 265, 267–273, 275, 277, 279, 281, 282, 284, 287, 290, 292, 294, 295, 298, 299, 301, 304, 307, 309, 311, 313, 314, 316, 318, 320, 322, 324, 326, 329, 332, 334, 336, 339, 342, 345, 348, 351, 354, 357, 360, 362	150
<code>\glsfirstlongfootnotefont</code>	153, 216, 442, 443
.... 258–263, 284–289, 301–305, 325–330	
<code>\glsfirstlonghyphenfont</code>	80, 81, 87, 102, 104, 115, 182, 205, 208–211, 215, 223, 226, 227, 368–378
344–357	
<code>\glsfirstlongonlyfont</code>	205
359–364	
<code>\glsfirstlonguserfont</code>	205
332–343	
<code>\GLSfirstplural</code>	101, 102, 192, 194, 195, 203, 204, 231–233
374	
<code>\Glsfirstplural</code>	86
374	
<code>\glsfirstplural</code>	204
374	
<code>\glsfirstpluralaccessdisplay</code>	79, 86, 88–92, 94–98, 226, 227, 236–248, 418–420
186	
<code>\GLSfmtname</code>	77, 86, 127, 181
60	
<code>\Glsfmtname</code>	204
59	
<code>\glsfmtname</code>	54
58–60	
<code>\GLSfmttext</code>	75, 76
60	
<code>\Glsfmttext</code>	500
59	
<code>\glsfmttext</code>	500
58–60	
<code>\glsforeachincategory</code>	79, 86, 94–98, 236–249, 344, 351, 353, 356, 358
250	
<code>\glsgenentryfmt</code>	134, 231, 233
77	
<code>\glsgetattribute</code>	7, 8, 36, 37, 56, 77, 80–84, 101–103, 108, 109, 137, 177, 178, 182, 224–226, 247–249, 261, 263, 287, 289, 303, 305, 328, 330, 333, 335, 337–339, 341, 351, 353, 356, 358, 420
.... 84, 108, 124, 128–130, 153, 177, 182, 208, 209, 211–214, 216, 221, 421–424	
<code>\glsgetcategoryattribute</code>	134, 231, 233, 253, 255– 264, 266, 268, 269, 271–277, 279, 281, 284, 286, 287, 289–295, 297, 301, 303– 314, 316, 318, 320, 321, 323, 325–328, 330, 332, 333, 335, 337–343, 345–347, 349, 351, 353–356, 358, 359, 361, 362, 364
202	
<code>\glsgetgroupitle</code>	217
489, 491, 492	
<code>\glsgetwidestname</code>	135, 450
508	
<code>\glsgroupheading</code>	
.... 172, 490–499, 501, 502, 504–507, 518–525, 531, 539, 540, 542, 543, 545, 546, 548, 549, 551, 553, 554, 556, 557, 561	
<code>\glsgroupskip</code>	
172, 490, 493–500, 502, 505, 506, 519, 531, 539, 541–545, 547, 548, 550, 551, 553, 554, 556, 558, 562	
<code>\glshasattribute</code>	
.. 83, 108, 123, 124, 128, 129, 131, 132, 153, 177, 182, 208, 209, 211, 212, 214, 216, 221, 253, 255–257, 259, 261–263, 266, 268, 269, 271–276, 284, 286, 287, 289–293, 301, 303–305, 307–310, 312, 314, 321, 325–328, 330, 332, 333, 335,	
	337–339, 341–343, 345–347, 349, 351, 353–356, 358, 359, 361, 362, 364, 421–424
<code>\glshascategoryattribute</code>	203, 409
<code>\glshex</code> ..	456–462, 464–470, 472–475, 477–485
<code>\glshyperlink</code>	109, 440
<code>\glshypernavsep</code>	150
<code>\glshypernumber</code>	153, 216, 442, 443
<code>\glsifattribute</code>	80, 81, 87, 102, 104, 115, 182, 205, 208–211, 215, 223, 226, 227, 368–378
80, 81, 87, 102, 104, 115, 182, 205, 208–211, 215, 223, 226, 227, 368–378	
<code>\glsifcategory</code>	205
<code>\glsifcategoryattribute</code>	101, 102, 192, 194, 195, 203, 204, 231–233
101, 102, 192, 194, 195, 203, 204, 231–233	
<code>\glsifnotregular</code>	86
<code>\glsifnotregularcategory</code>	204
<code>\glsifplural</code>	79, 86, 88–92, 94–98, 226, 227, 236–248, 418–420
88–92, 94–98, 226, 227, 236–248, 418–420	
<code>\glsifregular</code>	77, 86, 127, 181
<code>\glsifregularcategory</code>	204
<code>\glsifusetranslator</code>	54
<code>\glsignore</code>	75, 76
<code>\glsinlinedescformat</code>	500
<code>\glsinlinesubdescformat</code>	500
<code>\glsinsert</code>	79, 86, 94–98, 236–249, 344, 351, 353, 356, 358
86, 94–98, 236–249, 344, 351, 353, 356, 358	
<code>\glskeylisttok</code>	134, 231, 233
<code>\glslabel</code>	7, 8, 36, 37, 56, 77, 80–84, 101–103, 108, 109, 137, 177, 178, 182, 224–226, 247–249, 261, 263, 287, 289, 303, 305, 328, 330, 333, 335, 337–339, 341, 351, 353, 356, 358, 420
7, 8, 36, 37, 56, 77, 80–84, 101–103, 108, 109, 137, 177, 178, 182, 224–226, 247–249, 261, 263, 287, 289, 303, 305, 328, 330, 333, 335, 337–339, 341, 351, 353, 356, 358, 420	
<code>\glslabeltok</code>	134, 231, 233, 253, 255– 264, 266, 268, 269, 271–277, 279, 281, 284, 286, 287, 289–295, 297, 301, 303– 314, 316, 318, 320, 321, 323, 325–328, 330, 332, 333, 335, 337–343, 345–347, 349, 351, 353–356, 358, 359, 361, 362, 364
<code>\glsletentryfield</code>	217
<code>\glslink</code>	135, 450
<code>\glslink options</code>	
counter	9
format	215
hyper	364
hyperoutside	80, 81
noindex	8, 102, 364
textformat	83
theHvalue	83
thevalue	83, 174
wrgloss	7, 80

<code>\glslinkcheckfirsthyperhook</code>	102	<code>\glslongextraLocationDescNameTabularHeader</code>	
<code>\glslinkpostsetkeys</code>	9, 83, 178	542, 543
<code>\glslinkpresetkeys</code>	9, 83, 178	<code>\glslongextraLocationDescSymNameHeader</code>	
<code>\glslinkvar</code>	105	556
<code>\glslinkwrcontent</code>	84, 178	<code>\glslongextraLocationDescSymNameTabularFooter</code>	
<code>\glslistchildpostlocation</code>	490	555
<code>\glslistchildprelocation</code>	490, 491	<code>\glslongextraLocationDescSymNameTabularHeader</code>	
<code>\glslistdesc</code>	490, 491	555
<code>\glslistdottedwidth</code>	488	<code>\glslongextraLocationFmt</code>	
<code>\glslistexpandedname</code>	489	540, 543, 546, 550, 553, 556
<code>\glslistgroupafterheader</code>	491, 492	<code>\glslongextraLocationSymDescNameHeader</code>	
<code>\glslistgroupheaderfmt</code>	489, 491, 492	553
<code>\glslistgroupheaderitem</code>	491, 492	<code>\glslongextraLocationSymDescNameTabularFooter</code>	
<code>\glslistgroupskip</code>	490	551, 552
<code>\glslistinit</code>	490	<code>\glslongextraLocationSymDescNameTabularHeader</code>	
<code>\glslistitem</code>	490	551, 552
<code>\glslistnavigationitem</code>	491, 492	<code>\glslongextraLocSetDescWidth</code> ..	540, 543
<code>\glslistprelocation</code>	489–491	<code>\glslongextraNameAlign</code>	
<code>\glslocalreset</code>	403	538–549, 551, 552, 554–556
<code>\glslocalunset</code>	102, 402, 403	<code>\glslongextraNameDescHeader</code>	539
<code>\glslongaccessdisplay</code>	190	<code>\glslongextraNameDescLocationHeader</code>	540
<code>\glslongdefaultfont</code>	235, 254, 256, 258, 265, 267, 270, 273, 275, 277, 279, 281–283, 290, 292, 294, 295, 297–299, 307, 311, 314, 316, 318, 322, 331, 345, 359	<code>\glslongextraNameDescLocationTabularFooter</code>	
<code>\glslonggemfont</code> 306, 309, 312, 313, 320, 323, 324		539, 540
<code>\glslongextraDescAlign</code>		<code>\glslongextraNameDescLocationTabularHeader</code>	
.....	538–549, 551, 552, 554–556	539, 540
<code>\glslongextraDescFmt</code> 535, 539, 540, 542,		<code>\glslongextraNameDescSymHeader</code>	545
543, 545, 546, 548, 550, 551, 553, 554, 556		<code>\glslongextraNameDescSymLocationHeader</code>	
<code>\glslongextraDescNameHeader</code>	542	546
<code>\glslongextraDescNameTabularFooter</code> 541		<code>\glslongextraNameDescSymLocationTabularFooter</code>	
<code>\glslongextraDescNameTabularHeader</code>		545, 546
.....	541, 542	<code>\glslongextraNameDescSymTabularFooter</code>	
<code>\glslongextraDescSymNameHeader</code>	554	544
<code>\glslongextraDescSymNameTabularFooter</code>		<code>\glslongextraNameDescSymTabularHeader</code>	
.....	553, 554	544
<code>\glslongextraDescSymNameTabularHeader</code>		<code>\glslongextraNameDescTabularFooter</code>	
.....	553, 554	536, 539
<code>\glslongextraGroupHeading</code> 539, 540, 542,		<code>\glslongextraNameDescTabularHeader</code>	
543, 545, 546, 548, 549, 551, 553, 554, 556		536, 539
<code>\glslongextraHeaderFmt</code> . 536, 537, 539,		<code>\glslongextraNameFmt</code> ... 539, 540, 542,	
541, 542, 544, 545, 547–550, 552, 553, 555		543, 545, 546, 548, 550, 551, 553, 554, 556	
<code>\glslongextraLocationAlign</code>		<code>\glslongextraNameSymDescHeader</code>	548
.....	540, 543, 546, 549, 552, 555, 556	<code>\glslongextraNameSymDescLocationHeader</code>	
<code>\glslongextraLocationDescNameHeader</code> 543		549
<code>\glslongextraLocationDescNameTabularFooter</code>		<code>\glslongextraNameSymDescLocationTabularFooter</code>	
.....	542, 543	548, 549
		<code>\glslongextraNameSymDescLocationTabularHeader</code>	
		548, 549

<code>\glslongextraNameSymDescTabularFooter</code>	333, 335, 337, 338, 340–343, 345–347, 349, 350, 352, 354, 355, 359, 361, 362, 364
<code>\glslongextraNameSymDescTabularHeader</code>	547, 548
<code>\glslongextraSetDescWidth</code>	537–539, 541, 542
<code>\glslongextraSubDescFmt</code>	539, 541–543, 545, 547, 548, 550, 551, 553, 554, 556
<code>\glslongextraSubLocationFmt</code>	541, 543, 547, 550, 553, 556
<code>\glslongextraSubNameFmt</code>	539, 541–543, 545, 546, 548, 550, 551, 553, 554, 556
<code>\glslongextraSubSymbolFmt</code>	545, 547, 548, 550, 551, 553, 554, 556
<code>\glslongextraSymbolAlign</code>	544–552, 554–556
<code>\glslongextraSymbolFmt</code>	535, 545, 546, 548, 550, 551, 553, 554, 556
<code>\glslongextraSymDescNameHeader</code>	551
<code>\glslongextraSymDescNameTabularFooter</code>	550, 551
<code>\glslongextraSymDescNameTabularHeader</code>	550, 551
<code>\glslongextraSymLocSetDescWidth</code>	546, 549, 552, 555
<code>\glslongextraSymSetDescWidth</code>	538, 544, 547, 548, 550, 551, 554
<code>\glslongextraTabularVAlign</code>	538, 540, 541, 543, 544, 546, 547, 549, 550, 552, 554, 555
<code>\glslongextraUpdateWidest</code>	445–447
<code>\glslongextraUpdateWidestChild</code>	445–447
<code>\GlsLongExtraUseTabularfalse</code>	538
<code>\glslongfont</code>	112, 235, 241, 242, 245–247, 254, 256, 257, 259, 260, 262, 265–267, 269, 270, 272, 273, 275, 277, 279, 281, 282, 284, 287, 290, 292, 294, 295, 298, 299, 301, 304, 307, 309, 311, 313, 314, 316, 318, 320, 322, 324, 326, 329, 332, 334, 336, 339, 342, 345, 348, 351, 354, 357, 360–362
<code>\glslongfootnotefont</code>	258, 259, 262, 284, 287, 301, 304, 326, 329
<code>\glslonghyphenfont</code>	345, 347–349, 351, 354, 356, 357
<code>\glslongonlyfont</code>	359, 360, 362
<code>\glslongpltok</code>	233, 253, 255–258, 260, 269, 272–276, 281, 284, 286, 290–293, 297, 301, 303, 306, 308, 310, 312, 314, 318, 320, 323, 325, 327, 332,
<code>\glslongpluralaccessdisplay</code>	190
<code>\glslongtok</code>	134, 231, 233, 253–258, 260, 261, 264, 266, 269, 272–277, 280, 281, 284, 286, 290–294, 297, 301, 303, 306–310, 312–314, 318, 320, 323, 325, 327, 328, 332, 333, 335, 337–343, 345–347, 349–352, 354–356, 359, 361, 362, 364
<code>\glslonguserfont</code>	331, 332, 334, 336, 337, 339, 340, 342
<code>\glsmcols</code>	521, 522, 524, 525
<code>\GLSname</code>	370, 371
<code>\Glsname</code>	370, 371
<code>\glsname</code>	370
<code>\glsnameaccessdisplay</code>	183, 184, 210, 212
<code>\glsnamefont</code>	210–213, 215, 537
<code>\glsnavhyperlink</code>	150
<code>\glsnavhyperlinkname</code>	106, 107
<code>\glsnavhypertarget</code>	490–492, 502, 505, 507, 520–526
<code>\glsnavigation</code>	490–492, 502, 505, 507, 520–525
<code>\glsnextpages</code>	143, 168
<code>\glsnoidxdisplayloc</code>	148, 442
<code>\glsnoidxdisplaylocclsthandler</code>	148
<code>\glsnoidxlocclst</code>	149, 173
<code>\glsnoidxnumberlistloophandler</code>	148
<code>\glsnonextpages</code>	143, 168
<code>\glsnonumberlistfalse</code>	74
<code>\glsnonumberlisttrue</code>	75
<code>\glsnopostdotfalse</code>	144
<code>\glsnopostdottrue</code>	144
<code>\glsnumberlistloop</code>	141
<code>\glsnumlistlastsep</code>	147, 438
<code>\glsnumlistsep</code>	147, 438
<code>\glsopenbrace</code>	62, 156–158
<code>\glsorder</code>	139, 162
<code>\glspagelistwidth</code>	493, 495, 497, 499, 536, 538
<code>\glspar</code>	171, 558
<code>\glspatchLToutput</code>	539, 540, 542–544, 546, 548, 549, 551, 552, 554, 555
<code>\glspdfmftfull</code>	386
<code>\glspdfmftfullpl</code>	387
<code>\glspenaltygroupskip</code>	539, 541, 542, 544, 545, 547, 548, 550, 551, 553, 554, 556
<code>\GLSpl</code>	121, 133, 177
<code>\Glspl</code>	72, 121, 133, 177
<code>\glspl</code>	71, 121, 133, 177

<code>\GLSplural</code>	372	290–295, 297, 301–303, 305–310, 312–
<code>\Glsplural</code>	372	314, 316, 318, 320, 325, 327, 328, 330,
<code>\glsplural</code>	372	332, 333, 335, 337–343, 345, 346, 349,
<code>\glspluralaccessdisplay</code>	185	350, 352, 354–356, 358, 359, 361, 362, 364
<code>\glspluralsuffix</code>	161, 231, 235	<code>\glsshowtargetfont</code>
<code>\glspostdescription</code> ...	18, 19, 144, 223,	29
	488, 489, 492–500, 503–506, 534, 559, 560	<code>\glsshowtargetinner</code>
<code>\glspostinline</code>	500	27–29
<code>\glspostlinkhook</code>		<code>\glsshowtargetinnersymleft</code>
	78–80, 94–98, 113, 236–246, 413, 414, 420	28, 29
<code>\glsprestandardsort</code>	141	<code>\glsshowtargetinnersymright</code>
<code>\glsreset</code>	403	28, 29
<code>\glsresetentrylist</code>	151, 166, 168	<code>\glsshowtargetouter</code>
<code>\glssee</code>	63–65	27, 28
<code>\glsseefirstitem</code>	62	<code>\glsshowtargetsymbols</code>
<code>\glsseeformat</code>	58, 61, 68, 140, 148	29
<code>\glsseeitem</code>	62	<code>\glssubentryitem</code>
<code>\glsseelastoxfordsep</code>	61	164,
<code>\glsseelastsep</code>	61, 62	488, 490–500, 502, 504, 506, 518, 531, 535
<code>\glsseelist</code>	61	<code>\glssymbolaccessdisplay</code>
<code>\glsseesep</code>	61	186, 187
<code>\glssetabbrfmt</code>		<code>\glssymbolpluralaccessdisplay</code>
	77, 86, 111, 112, 208, 209, 211,	187
	212, 214, 220, 221, 236–241, 243, 244, 247	<code>\glstarget</code> ...
<code>\glssetattribute</code>	253, 255–	163, 165, 488–500, 502, 504,
	257, 259, 261–264, 266, 268, 269, 271–	506, 518, 519, 530, 531, 534, 535, 558, 560
	277, 279, 281, 284, 286, 287, 289–295,	<code>\GLStext</code>
	297, 301, 303–305, 307–312, 314, 316,	371, 372
	318, 320, 321, 323, 325–328, 330, 332,	<code>\GlStext</code>
	333, 335, 337–339, 341–343, 345–347,	371
	349, 351, 353–356, 358, 359, 361, 362, 364	<code>\glstext</code>
<code>\glssetcategoriesattribute</code>	201	371
<code>\glssetcategoryattribute</code>	122, 133, 137,	<code>\glstextaccessdisplay</code>
	177, 183, 196, 197, 202, 203, 205, 206, 222	184
<code>\glssetnoexpandfield</code>	13	<code>\glstextformat</code>
<code>\glssettoctitle</code>	143, 167	80, 84
<code>\glssetwidest</code>	446	<code>\glstextup</code>
<code>\glsshortaccessdisplay</code>	189	273
<code>\glsshortaccsupp</code>	196	<code>\glstopic@postchildren</code>
<code>\glsshortpltok</code>	233, 253, 255–	561, 562
	258, 260, 261, 263, 264, 266, 273–278,	<code>\glstopic@prechildren</code>
	284, 286, 288, 290–295, 301, 303, 305,	557, 561, 562
	307–310, 312, 314, 316, 325, 327, 328,	<code>\glstopic@prevlevel</code>
	330, 332, 333, 335, 337–343, 345, 346,	557, 561, 562
	351, 352, 354–356, 358, 359, 361, 362, 364	<code>\glstopicAssignSubIndent</code>
<code>\glsshortpluralaccessdisplay</code> ..	189, 190	558, 562
<code>\glsshorttok</code>		<code>\glstopicAssignWidest</code>
	134, 231–233, 253–258, 260, 261, 263,	559
	264, 266, 271–278, 280, 284, 286, 288,	<code>\glstopicCols</code>
		562
		<code>\glstopicColsEnv</code>
		562
		<code>\glstopicDesc</code>
		558
		<code>\glstopicGroupHeading</code>
		557, 561
		<code>\glstopicInit</code>
		557, 561
		<code>\glstopicItem</code>
		557, 562
		<code>\glstopicLoc</code>
		558
		<code>\glstopicMarker</code>
		558
		<code>\glstopicMidSkip</code>
		558
		<code>\glstopicParIndent</code>
		557, 562
		<code>\glstopicPostSkip</code>
		558
		<code>\glstopicPreSkip</code>
		558
		<code>\glstopicSubIndent</code>
		559
		<code>\glstopicSubItem</code>
		558, 562
		<code>\glstopicSubItemBox</code>
		560
		<code>\glstopicSubItemhangindent</code>
		559
		<code>\glstopicSubItemParIndent</code>
		559
		<code>\glstopicSubItemSep</code>
		560
		<code>\glstopicSubLoc</code>
		560
		<code>\glstopicSubNameFont</code>
		560
		<code>\glstopicSubPreLocSep</code>
		560
		<code>\glstopicTitle</code>
		558

<code>\glstopicTitleFont</code>	558	<code>\glxtr@dblfloat</code>	18
<code>\glstopicwidest</code>	559–561	<code>\glxtr@do@alsoindex@wrglossary</code>	16
<code>\glstreechilddesc</code>	504	<code>\glxtr@do@autoadd</code>	8, 82, 83
<code>\glstreeChildDescLoc</code>	502, 505	<code>\glxtr@doooption</code>	4, 18, 19, 26–28, 31
<code>\glstreechildpredesc</code>	503	<code>\glxtr@endbookindex</code>	530
<code>\glstreechildprelocation</code>	504, 506	<code>\glxtr@fields</code>	160
<code>\glstreechildsymbol</code>	502, 505	<code>\glxtr@float</code>	18
<code>\glstreedefaultnamefmt</code>	501	<code>\glxtr@grptitle</code> ...	502, 505, 507, 519–526
<code>\glstreedesc</code>	503	<code>\glxtr@headentry@p</code>	115, 116
<code>\glstreeDescLoc</code>	502, 504, 508	<code>\glxtr@hyperoutsidfalse</code>	81
<code>\glstreegroupheaderfmt</code>		<code>\glxtr@hyperoutsidettrue</code>	80, 81
.....	502, 505, 507, 519–526, 528	<code>\glxtr@ifnextpunc</code>	228
<code>\glstreegroupheaderskip</code>		<code>\glxtr@ifpunctoken</code>	228
.....	502, 503, 505, 507, 519–526	<code>\glxtr@inc@linkcount</code>	83, 183
<code>\glstreegroupskip</code> ..	501, 502, 505, 506, 519	<code>\glxtr@inc@wrglossaryctr</code>	7, 8, 26, 32
<code>\glstreeindent</code>	504, 506, 517–519	<code>\glxtr@indexonly@saveentrycounter</code>	15, 16, 32
<code>\glstreeitem</code>	501, 521, 530, 531	<code>\glxtr@keylist</code>	70–72
<code>\glstreenamebox</code>	518, 519	<code>\glxtr@label</code>	533
<code>\glstreenamefmt</code>		<code>\glxtr@langtag</code>	161
.....	500, 502, 504, 506, 508, 511–519	<code>\glxtr@linkprefix</code>	160, 161
<code>\glstreenavigationfmt</code>	502, 505, 507, 520–525	<code>\glxtr@loaddialect</code>	437, 486
<code>\glstreeNoDescSymbolPreLocation</code>	503, 504	<code>\glxtr@locationhypertext</code>	443, 444
<code>\glstreenonamechilddesc</code>	506	<code>\glxtr@makeglossaries</code>	139
<code>\glstreenonameChildDescLoc</code>	506	<code>\glxtr@mgl@applyopts</code>	409
<code>\glstreenonamedesc</code>	506	<code>\glxtr@mgl@checklastelement</code>	413
<code>\glstreenonameDescLoc</code>	506	<code>\glxtr@mgl@inner</code>	426
<code>\glstreenonamesymbol</code>	506	<code>\glxtr@newabbreviation</code>	136, 230
<code>\glstreepredesc</code>	503, 505	<code>\glxtr@newmgl</code>	428–436
<code>\glstreePreHeader</code> ..	502, 505, 507, 519–525	<code>\glxtr@next</code>	228
<code>\glstreeprelocation</code>	501, 503, 506	<code>\glxtr@org@@do@wrglossary</code>	32
<code>\glstreesubitem</code>	501, 531	<code>\glxtr@org@dohyperlink</code>	106, 402
<code>\glstreesubsubitem</code>	501, 531	<code>\glxtr@org@getgrouptitle</code>	150
<code>\glstreesymbol</code>	502, 504	<code>\glxtr@org@newignoredglossary</code>	52
<code>\GlsrLetField</code>	45	<code>\glxtr@org@makenoidxglossaries</code>	68
<code>\glstype</code>	79, 82, 94–98, 178, 236–246	<code>\glxtr@pluralsuffixes</code>	160, 161
<code>\glunset</code>	66, 102, 124–126, 402, 403	<code>\glxtr@printgloss@checkexists</code>	144
<code>\glupdatewidest</code>	445	<code>\glxtr@printgloss@groupstrue</code>	146
<code>\gluserdescription</code>	332, 333, 335, 339, 342	<code>\glxtr@process</code>	166, 169, 170
<code>\glswrite</code>	62, 139	<code>\glxtr@provideignoredglossary</code>	54
<code>\glswriteentry</code>	7, 8	<code>\glxtr@punclist</code>	227, 228
<code>\Glsxtr</code>	72	<code>\glxtr@record</code>	10, 11, 161
<code>\glxtr</code>	72	<code>\glxtr@record@nameref</code>	11, 161
<code>\glxtr@do@wrglossary</code>	7, 9, 12, 15	<code>\glxtr@record@nr</code>	15
<code>\glxtr@addloclistfield</code>	14, 16	<code>\glxtr@recordsee</code>	12
<code>\glxtr@addunused</code>	66	<code>\glxtr@renewcommand</code>	439
<code>\glxtr@applyabbrvfmt</code>	246, 247	<code>\glxtr@resource</code>	159, 160
<code>\glxtr@applyabbrvstyle</code>	231, 250	<code>\glxtr@s@newignoredglossary</code>	52
<code>\glxtr@beginbookindex</code>	529	<code>\glxtr@s@provideignoredglossary</code> ...	54
<code>\glxtr@counterrecord</code>	163		

<code>\glxtr@saveentrycounter</code>	7, 9, 12, 103	<code>\glxtrautoindexassignsort</code>	217
<code>\glxtr@setaccessdisplay</code>	214	<code>\glxtrautoindexentry</code>	217
<code>\glxtr@setbookindexmark</code>	532	<code>\glxtrautoindexesc</code>	217
<code>\glxtr@setup@docurrent</code>	413	<code>\glxtrbibaddress</code>	441
<code>\glxtr@setup@record</code>	14–16, 31, 33	<code>\glxtrbibauthor</code>	441
<code>\glxtr@shortcutsval</code>	160, 161	<code>\glxtrbibbooktitle</code>	441
<code>\glxtr@texencoding</code>	161	<code>\glxtrbibchapter</code>	441
<code>\glxtr@undefaction@nr</code>	6	<code>\glxtrbibedition</code>	441
<code>\glxtr@undefaction@val</code>	6	<code>\glxtrbibhowpublished</code>	441
<code>\glxtr@usesee</code>	58	<code>\glxtrbibinstitution</code>	441
<code>\glxtr@warnonexistsordo</code> .	6, 14–16, 56, 57	<code>\glxtrbibjournal</code>	441
<code>\glxtr@writefields</code>	159	<code>\glxtrbibmonth</code>	441
<code>\glxtrabbrvfont</code>	77	<code>\glxtrbibnote</code>	441
<code>\glxtrabbrvfootnote</code> ...	258–261, 263, 284–287, 289, 301–303, 305, 325–328, 330	<code>\glxtrbibnumber</code>	441
<code>\glxtrabbrvpluralsuffix</code> ...	161, 194, 236, 254, 256, 259, 262, 264, 267, 270, 273, 289, 306, 331, 345, 347, 351, 357, 359	<code>\glxtrbiborganization</code>	441
<code>\glxtrabbrvtype</code>	20, 233	<code>\glxtrbibpages</code>	441
<code>\glxtrAccSuppAbbrSetFirstLongAttrs</code>	253, 255, 273, 275, 290, 292, 306, 308, 310, 312, 332, 333, 335, 338, 342, 345, 350, 353, 356, 359, 362	<code>\glxtrbibpublisher</code>	441
<code>\glxtrAccSuppAbbrSetNameLongAttrs</code>	260, 263, 278, 286, 288, 293, 302, 305, 327, 330	<code>\glxtrbibschooll</code>	441
<code>\glxtrAccSuppAbbrSetNameShortAttrs</code>	271, 297, 318, 319, 321, 349	<code>\glxtrbibseries</code>	441
<code>\glxtrAccSuppAbbrSetNoLongAttrs</code> 258, 261, 264, 266, 277, 284, 286, 295, 301, 303, 314, 316, 325, 328	<code>\glxtrbibtitle</code>	441
<code>\glxtrAccSuppAbbrSetTextShortAttrs</code>	255, 257, 274, 276, 291, 293, 308, 309, 311, 313, 337, 338, 340, 341, 343, 346, 352, 355, 358, 361, 363	<code>\glxtrbibtype</code>	441
<code>\glxtractivatenopost</code>	143	<code>\glxtrbibvolume</code>	441
<code>\glxtraddallcrossrefs</code>	65	<code>\glxtrbookindexatendgroup</code>	530
<code>\glxtralias</code>	103	<code>\glxtrbookindexatsubendgroup</code>	531
<code>\glxtrAltTreeIndent</code>	508	<code>\glxtrbookindexatsubsubendgroup</code> ..	531
<code>\glxtralttreeInit</code>	518, 524, 525	<code>\glxtrbookindexbetween</code>	530
<code>\glxtrAltTreePar</code>	508	<code>\glxtrbookindexbookmark</code>	532
<code>\glxtrAltTreeSetHangIndent</code> ...	508, 518	<code>\glxtrbookindexbookmarkprefix</code>	532
<code>\glxtrAltTreeSetSubHangIndent</code>	519	<code>\glxtrbookindexcols</code>	529, 530
<code>\glxtralttreeSubSymbolDescLocation</code>	519	<code>\glxtrbookindexcolspread</code>	529
<code>\glxtralttreeSymbolDescLocation</code> 508, 518	<code>\glxtrbookindexfirstmarkfmt</code>	533
<code>\glxtrassignactualsetup</code>	193	<code>\glxtrbookindexformatheader</code>	532
<code>\glxtrassignfieldfont</code>	87–94	<code>\glxtrbookindexgroupskip</code>	532
<code>\glxtrautoindex</code>	216	<code>\glxtrbookindexlastmarkfmt</code>	533
		<code>\glxtrbookindexlocation</code>	528, 530
		<code>\glxtrbookindexmulticolenv</code> ..	529, 530
		<code>\glxtrbookindexname</code>	527, 530
		<code>\glxtrbookindexparentchildsep</code>	528–530
		<code>\glxtrbookindexparentsubchildsep</code> 529–531
		<code>\glxtrbookindexprelocation</code> ...	527, 530
		<code>\glxtrbookindexsubbetween</code>	531
		<code>\glxtrbookindexsublocation</code>	531
		<code>\glxtrbookindexsubname</code>	531
		<code>\glxtrbookindexsubprelocation</code>	531
		<code>\glxtrbookindexsubsubbetween</code>	531
		<code>\glxtrbookindexthepage</code>	532, 533
		<code>\glxtrcat</code>	71, 72
		<code>\glxtrchecknohyperfirst</code>	87–89

\glxtrcombingdiacriticIIIrules .	457	\glxtrfootnotedesname	
\glxtrcombingdiacriticIIrules ..	457	260, 263, 286, 288, 302, 305, 327, 330
\glxtrcombingdiacriticIrules ...	457	\glxtrfootnotedesort	
\glxtrcombingdiacriticIVrules ..	457	260, 263, 286, 288, 302, 305, 327, 330
\glxtrComputeTreeIndent	518, 519	\glxtrfootnotename	
\glxtrComputeTreeSubIndent	519	258, 261, 284, 286, 301, 303, 325, 328
\glxtrcounterprefix	152	\GlsXtrForeignTextField	47
\glxtrcurrencyrules	460	\GlsXtrFormatLocationList	74, 77, 515–517
\glxtrcurrentgrptitle	532	\GLSxtrfull	21, 99, 376–378
\glxtrcurrentmglsname	426	\Glsxtrfull	20, 21, 99, 377
\GlsXtrDefaultResourceOptions	159	\glxtrfull	20, 21, 98, 376, 377
\Glsxtrdefaultsubsequentfmt ...	249, 251	\Glsxtrfullformat	
\glxtrdefaultsubsequentfmt ...	249, 251	.	234, 248, 251, 254, 256, 259, 262, 265,
\Glsxtrdefaultsubsequentplfmt .	249, 251		267, 271, 274, 276, 278, 279, 282, 283,
\glxtrdefaultsubsequentplfmt .	249, 251		285, 287, 290, 292, 295, 296, 299–301,
\GlsXtrDefineAbbreviationShortcuts			304, 307, 309, 311, 313, 315, 317, 319,
.....	22, 23		321, 323, 325, 326, 329, 333, 334, 336,
\GlsXtrDefineAcShortcuts	23		339, 343, 346, 348, 352, 354, 357, 360, 363
\GlsXtrDefineOtherShortcuts	23	\glxtrfullformat	234,
\glxtrdetoklocation	176		248, 249, 251, 254, 256, 259, 262, 265,
\glxtrdiscardperiod	225		267, 270, 274, 275, 278, 279, 282–284,
\glxtrdisplayendloc	151		287, 290, 292, 294, 296, 298, 300, 301,
\glxtrdisplayendlohook	152		304, 307, 309, 311, 313, 315, 316, 319,
\glxtrdisplaysingleloc	151, 152		321, 323, 324, 326, 329, 332, 334, 336,
\glxtrdisplaystartloc	151		339, 342, 346, 348, 351, 354, 357, 360, 362
\glxtrdoautoindexname	104, 105, 213	\GLSxtrfullpl	21, 99, 377, 378
\glxtrdohyperlink	106, 109, 110	\Glsxtrfullpl	20, 21, 99, 378
\glxtrdopostpunc		\glxtrfullpl	20, 21, 99, 377
....	261, 263, 287, 289, 303, 305, 328, 330	\Glsxtrfullplformat	235,
\glxtrdowrglossaryhook	104, 105		248, 251, 254, 256, 259, 262, 265, 267,
\GlsXtrDualField	440		271, 274, 276, 278, 279, 282, 283, 285,
\glxxtremsuffix	307, 309, 311,		287, 291, 293, 295, 296, 299, 300, 302,
	313, 314, 316, 318, 320, 322, 323, 326, 329		304, 307, 309, 311, 313, 315, 317, 319,
\GlsXtrEnableEntryCounting	133		321, 323, 325, 326, 329, 333, 334, 336,
\GlsXtrEnableEntryUnitCounting	121, 122		339, 343, 346, 348, 352, 354, 357, 360, 363
\GlsXtrEnableOnTheFly	70, 72, 73	\glxtrfullplformat	
\glxxtrendfor	39	248, 251, 254, 256, 259, 262, 265,
\glxtrfieldlistgadd	162		267, 271, 274, 276, 278, 279, 282, 283,
\glxtrfieldtitlecase	208–211, 215		285, 287, 290, 292, 295, 296, 299–301,
\glxtrfieldtitlecasecs	207		304, 307, 309, 311, 313, 315, 317, 319,
\glxtrfieldxifinlist	171		321, 323, 324, 326, 329, 332, 334, 336,
\glxtrfirstscfont	273		339, 342, 346, 348, 351, 354, 357, 360, 362
\glxtrfirstsmfont	289	\glxtrfullsep	234,
\GlsXtrFmtDefaultOptions	36		253–257, 259, 260, 262, 263, 265–270,
\glxtrfmtdisplay	36		273–283, 285, 288, 290–298, 300, 302,
\glxtrfmtexternalnameref	443, 444		304–317, 319–324, 326, 327, 329, 331,
\GlsXtrFmtField	36, 37		344–346, 348, 350, 353–356, 360, 361, 363
\glxtrfmtinternalnameref	443, 444	\glxtrgenabbrvfmt	77
		\glxtrgeneralpuncIIrules	460

<code>\glxstrgeneralpuncIrules</code>	460	<code>\glxstrindexaliased</code>	103, 104
<code>\glxstrgetgroupTitle</code>		<code>\glxstrindexseealso</code>	64, 65
.....	150, 502, 505, 507, 519–525, 532	<code>\glxstrinithyperoutside</code>	83
<code>\glxstrgroupfield</code>	172	<code>\glxstrinitwrgloss</code>	83, 178
<code>\Glsxtrheadfirst</code>	367	<code>\glxstrinitwrglossbeforefalse</code>	80
<code>\glxstrheadfirst</code>	367	<code>\glxstrinitwrglossbeforetrue</code>	80
<code>\Glsxtrheadfirstplural</code>	367	<code>\Glsxtrinlinefullformat</code>	
<code>\glxstrheadfirstplural</code>	367	235, 237, 251, 260,
<code>\Glsxtrheadfull</code>	367		263, 265, 267, 268, 270, 277, 279–281,
<code>\glxstrheadfull</code>	367		283, 285, 288, 294, 296–298, 300, 302,
<code>\Glsxtrheadfullpl</code>	367		305, 315–317, 319, 321, 322, 324, 327,
<code>\glxstrheadfullpl</code>	367		329, 334, 336, 340, 348, 352, 358, 360, 363
<code>\Glsxtrheadlong</code>	367	<code>\glxtrinlinefullformat</code>	
<code>\glxstrheadlong</code>	367	235–237, 251, 259, 262,
<code>\Glsxtrheadlongpl</code>	367		265, 267, 268, 270, 277, 279–281, 283,
<code>\glxstrheadlongpl</code>	367		285, 288, 294, 295, 297, 298, 300, 302,
<code>\Glsxtrheadname</code>	367		304, 314, 316–318, 320, 322, 324, 326,
<code>\glxstrheadname</code>	163, 367		329, 334, 336, 339, 348, 352, 357, 360, 363
<code>\Glsxtrheadplural</code>	367	<code>\Glsxtrinlinefullplformat</code>	
<code>\glxstrheadplural</code>	367	235, 238, 251, 260,
<code>\Glsxtrheadshort</code>	367		263, 265, 267, 268, 270, 278–280, 282,
<code>\glxstrheadshort</code>	367		283, 285, 288, 294, 296–298, 300, 302,
<code>\Glsxtrheadshortpl</code>	367		305, 315–317, 319, 321, 323, 324, 327,
<code>\glxstrheadshortpl</code>	367		329, 334, 336, 340, 348, 352, 358, 360, 363
<code>\Glsxtrheadtext</code>	367	<code>\glxtrinlinefullplformat</code>	
<code>\glxstrheadtext</code>	367	234, 235, 238, 239, 251, 259,
<code>\glxstrhiernamesep</code>	59, 60		262, 265, 267, 268, 270, 277, 279–281,
<code>\glxstrhyperlink</code>	25, 26, 108		283, 285, 288, 294, 296–298, 300, 302,
<code>\glxstrhyphensuffix</code>	345, 354		304, 315–317, 319, 320, 322, 324, 327,
<code>\glxstridentifyglslike</code>	174		329, 334, 336, 340, 348, 352, 357, 360, 363
<code>\glxstrifcounttrigger</code>	124–126	<code>\glxstrinsertinsidefalse</code>	253
<code>\glxstrifcustomdiscardperiod</code>	225	<code>\GlsXtrInternalLocationHyperlink</code>	26, 153
<code>\glxstrifemptyglossary</code> ..	151, 157, 166, 169	<code>\glxstrLatinA</code>	462–467
<code>\GlsXtrIfFieldEqNum</code>	164	<code>\glxstrLatinAELigature</code>	464, 466, 467
<code>\glxstrifhasfield</code> ..	47, 59, 60, 433, 440, 527	<code>\glxstrLatinE</code>	462–467
<code>\glxstrifhyphenstart</code>		<code>\glxstrLatinEszettSs</code>	463–465, 467
.....	344, 347, 349, 350, 353, 355, 356	<code>\glxstrLatinEszettSz</code>	463, 466
<code>\glxstrifindexing</code>	104	<code>\glxstrLatinEth</code>	462–466
<code>\glxstrifinmark</code>	85, 115–118, 366, 367	<code>\glxstrLatinH</code>	462–467
<code>\glxstrifmulti</code>	62, 66, 108, 397, 398, 427	<code>\glxstrLatinI</code>	462–467
<code>\glxstrifnextpunc</code>	228	<code>\glxstrLatinInsularG</code>	466
<code>\glxstrifperiod</code>	225, 227	<code>\glxstrLatinK</code>	462–467
<code>\glxstrifrecordtrigger</code>	178–180	<code>\glxstrLatinL</code>	462–467
<code>\GlsXtrIfUnusedOrUndefined</code>	104, 415	<code>\glxstrLatinM</code>	462–467
<code>\glxstrifwasfirstuse</code> ...	86–89, 94–98,	<code>\glxstrLatinN</code>	462–467
	101, 137, 226, 236, 239–246, 261–264,	<code>\glxstrLatinO</code>	462–467
	287–289, 303–306, 328, 330, 333, 335,	<code>\glxstrLatinOELigature</code>	464, 466, 467
	337–340, 351, 352, 356, 358, 417, 418, 420	<code>\glxstrLatinP</code>	462–467
<code>\glxstrinclinckcounter</code>	182	<code>\glxstrLatinS</code>	462–467

<code>\glxstrLatinT</code>	462–467	<code>\glxstrMathItalicUpsilon</code> ...	472, 476, 477
<code>\glxstrLatinThorn</code>	466	<code>\glxstrMathItalicXi</code>	471, 472, 476, 477
<code>\glxstrLatinX</code>	462–467	<code>\glxstrMathItalicZeta</code> ..	471, 472, 475, 476
<code>\GlsXtrLocationField</code>	172	<code>\glxstrmglsWarnAllSkipped</code>	410, 411
<code>\glxstrlocationhyperlink</code>	153	<code>\glxstrmglswrite</code>	419, 425
<code>\glxstrlocrangefmt</code>	152	<code>\GLSxtrmultientryadjustednamefmt</code> ..	452
<code>\GLSxtrlong</code>	21, 97, 374–376	<code>\GlsXtrmultientryadjustednamefmt</code> ..	451
<code>\Glsxtrlong</code>	20, 21, 97, 375, 376	<code>\Glsxtrmultientryadjustednamefmt</code> ..	451
<code>\glxtrlong</code>	20, 21, 96, 375	<code>\glxstrmultientryadjustednamefmt</code> ..	451
<code>\glxtrlonghyphen</code>	351, 352	<code>\GLSxtrmultientryadjustednameother</code> ..	452
<code>\glxtrlonghyphennohort</code>	348	<code>\GlsXtrmultientryadjustednameother</code> ..	451
<code>\glxtrlonghyphenshort</code>	346	<code>\Glsxtrmultientryadjustednameother</code> ..	451
<code>\glxtrlongnohortdescname</code> ..	269, 323, 347	<code>\glxstrmultientryadjustednameother</code> ..	451
<code>\glxtrlongnohortname</code>		<code>\glxstrmultientryadjustednamepostsep</code>	
.....	272, 280, 297, 318, 320, 349	451, 452
<code>\GLSxtrlongpl</code>	21, 98, 375, 376	<code>\glxstrmultientryadjustednamepresep</code>	
<code>\Glsxtrlongpl</code>	20, 21, 98, 376	451, 452
<code>\glxstrlongpl</code>	20, 21, 97, 375	<code>\glxstrmultientryadjustednamesep</code> ..	
<code>\glxtrlongshortdescname</code>	451–453
.....	255, 274, 291, 308, 310, 346, 352	<code>\glxstrmultilastotherindex</code>	406
<code>\glxtrlongshortdescsort</code>		<code>\glxstrmultimainindex</code>	406
....	255, 274, 291, 308, 310, 341, 346, 352	<code>\glxstrmultisupplcation</code>	442
<code>\glxtrlongshortname</code>		<code>\glxstrnamerefink</code>	442
.....	253, 273, 290, 306, 308, 332, 333, 345, 350	<code>\glxstrnewabbrevpresetkeyhook</code>	232
<code>\glxtrlongshortscuserdescname</code>	338	<code>\glxstrnewnumber</code>	22
<code>\glxtrlongshortscusername</code>	335	<code>\glxstrnewsymbol</code>	22
<code>\glxtrlongshortuserdescname</code> ..	337, 341	<code>\glxstrNoGlossaryWarning</code>	15, 24, 154
<code>\glxstrmarkhook</code>	365	<code>\GlsXtrNoGlsWarningAutoMake</code>	157
<code>\glxstrMathItalicAlpha</code> ..	471, 472, 475, 476	<code>\GlsXtrNoGlsWarningBuildInfo</code>	158
<code>\glxstrMathItalicBeta</code> ..	471, 472, 475, 476	<code>\GlsXtrNoGlsWarningCheckFile</code>	157
<code>\glxstrMathItalicChi</code>	472, 476, 477	<code>\GlsXtrNoGlsWarningEmptyMain</code> ..	157, 158
<code>\glxstrMathItalicDelta</code> ..	471, 472, 475, 476	<code>\GlsXtrNoGlsWarningEmptyNotMain</code> ...	157
<code>\glxstrMathItalicEpsilon</code> ..	471, 472, 475, 476	<code>\GlsXtrNoGlsWarningEmptyStart</code>	157
<code>\glxstrMathItalicEta</code> ...	471, 472, 475, 476	<code>\GlsXtrNoGlsWarningHead</code>	157
<code>\glxstrMathItalicGamma</code> ..	471, 472, 475, 476	<code>\GlsXtrNoGlsWarningMisMatch</code>	158
<code>\glxstrMathItalicIota</code> ..	471, 472, 475, 476	<code>\GlsXtrNoGlsWarningNoOut</code>	158
<code>\glxstrMathItalicKappa</code> ..	471, 472, 475, 476	<code>\GlsXtrNoGlsWarningTail</code>	158
<code>\glxstrMathItalicLambda</code> ..	471, 472, 475, 476	<code>\glxstrnopostpunc</code>	144
<code>\glxstrMathItalicMu</code>	471, 472, 475, 476	<code>\glxstronlydescname</code>	361, 363
<code>\glxstrMathItalicNu</code>	471, 472, 475, 477	<code>\glxstronlydescsort</code>	361, 363
<code>\glxstrMathItalicOmega</code>	472, 476, 477	<code>\glxstronlyname</code>	359
<code>\glxstrMathItalicOmicron</code> ..	471, 472, 476, 477	<code>\glxstronlysuffix</code>	360
<code>\glxstrMathItalicPhi</code>	472, 476, 477	<code>\glxstrorg@ifKV@glslink@hyper</code>	78
<code>\glxstrMathItalicPi</code>	471, 472, 476, 477	<code>\glxstrorglong</code>	193, 231, 254, 347
<code>\glxstrMathItalicPsi</code>	472, 476, 477	<code>\glxstrorgshort</code>	193, 231, 254
<code>\glxstrMathItalicRho</code> ...	471, 472, 476, 477	<code>\GLSxtrp</code>	115
<code>\glxstrMathItalicSigma</code>	472, 476, 477	<code>\Glsxtrp</code>	114
<code>\glxstrMathItalicTau</code>	472, 476, 477	<code>\glxstrp</code>	114, 116
<code>\glxstrMathItalicTheta</code> ..	471, 472, 475, 476		

\glxtrparen	226, 234, 253–257, 259, 260, 262, 263, 265–270, 273–283, 285, 288, 290–294, 296–298, 300, 302, 304–317, 319–324, 326, 327, 329, 331, 344–346, 348, 350, 353–356, 360, 361, 363	\glxtrscsuffix	273, 275, 277, 279, 281, 282, 284, 287, 335, 362
\glxtrpdfentryfmt	37	\glxtrscusersuffix	336
\GlsXtrPl	72	\GlsXtrSetActualChar	219
\glxtrPl	72	\glxtrsetaliasnoindex ...	15, 16, 103, 104
\glxtrpostdescription .	144, 205, 224, 500	\GlsXtrSetEncapChar	220
\glxtrposthyphenlong	356, 358	\GlsXtrSetEscChar	219
\glxtrposthyphenshort	351, 353	\glxtrsetfieldifexists	45
\glxtrposthyphensubsequent	351, 353, 356, 358	\glxtrsetglossarylabel	145
\glxtrpostlink	225	\GlsXtrSetLevelChar	219
\glxtrpostlinkendsentence	225	\glxtrsetpopts	113
\glxtrpostlinkhook	224	\glxtrsetupfulldefs	236– 239, 262, 264, 287, 289, 304, 306, 328, 330
\glxtrpostlocalreset	120, 123, 131	\GLSxtrshort	21, 95, 117, 118, 368, 369
\glxtrpostlocalunset	120, 123, 131	\Glsxtrshort	20, 21, 94, 369
\glxtrpostlongdescription	52	\glxtrshort	20, 21, 94, 368
\glxtrpostnamehook	211–213, 215	\glxtrshortdescname ...	266, 278, 295, 316
\GlsXtrPostNewAbbreviation	233, 251, 253, 255– 258, 261, 263, 264, 266, 268, 269, 271– 277, 279, 281, 284, 286–288, 290–295, 297, 301, 303, 305, 307–310, 312, 314, 316, 318, 320, 321, 323, 325–328, 330, 332, 333, 335, 337–343, 345–347, 349, 351, 352, 354–356, 358, 359, 361, 362, 364	\glxtrshorthyphen	357
\glxtrpostreset	120, 123, 131	\glxtrshorthyphenlong	354
\glxtrpostunset	119, 122, 130, 131	\glxtrshortlongdescname	257, 276, 293, 311, 313, 355, 358
\glxtrprelocation	489, 492, 494, 496, 498, 501, 527	\glxtrshortlongdescsort	257, 276, 293, 311, 313, 343, 355, 358
\glxtrprotectlinks	108, 109	\glxtrshortlongname	256, 275, 292, 310, 312, 338, 342, 354, 356
\glxtrprovideaccsuppcmd	196, 197	\glxtrshortlonguserdescname ..	340, 343
\GlsXtrRecordCounter	12	\glxtrshortnolongname .	264, 277, 294, 314
\glxtrrecordtriggervalue	177	\GLSxtrshortPl	21, 96, 368–370
\GlsXtrRecordWarning	158	\GlsxtrshortPl	20, 21, 95, 369, 370
\glxtrregularfont	77, 86	\glxtrshortPl	20, 21, 95, 368, 369
\glxtrresourcecount	160	\glxtrshowtargetinner	28, 29
\glxtrresourcefile	160	\glxtrshowtargetouter	28, 29
\glxtrresourceinit	159	\glxtrshowtargetsymbolleft	28, 29
\glxtrrestoremarkhook	365	\glxtrshowtargetsymbolright ...	28, 29
\glxtrrestorepostpunc	144, 145	\glxtrsmfont	289
\glxtrscfont	272	\glxtrsmsuffix	290, 292, 294, 295, 298, 299, 301, 304
\glxtrsconlydescname	364	\GlsXtrStandaloneEntryName	163
\glxtrsconlydescsort	364	\GlsXtrStandaloneEntryOther	165
\glxtrsconlyname	362	\GlsXtrStandaloneGlossaryType .	163, 164
\glxtrsconlysuffix	362	\GlsXtrStandaloneSubEntryItem .	163, 164
		\Glsxtrsubsequentfmt	248, 251, 270, 281, 283, 298, 299, 318, 320, 322, 324, 348, 351, 357
		\glxtrsubsequentfmt	247, 248, 251, 270, 281, 282, 298, 299, 318, 320, 322, 324, 348, 351, 357

<code>\Glsxtrsubsequentplfmt</code>	<code>\glsxtrunsrtdo</code>
..... 247, 251, 270, 281, 283,	170, 171
298, 299, 318, 320, 322, 324, 348, 351, 357	<code>\glsxtrUpAlpha</code>
<code>\glsxtrsubsequentplfmt</code>	470, 471, 475, 476
..... 247, 251, 270, 281, 282,	<code>\glsxtrUpBeta</code>
298, 299, 318, 320, 322, 324, 348, 351, 357	470, 471, 475, 476
<code>\glsxtrsuppllocationurl</code>	<code>\glsxtrUpChi</code>
153, 442–444	470, 471, 476, 477
<code>\glsxtrtagfont</code>	<code>\glsxtrUpDelta</code>
223	470, 471, 475, 476
<code>\GLSxtrtitlefirst</code>	<code>\glsxtrUpDigamma</code>
383	470, 471, 475
<code>\Glsxtrtitlefirst</code>	<code>\glsxtrUpEpsilon</code>
366–368, 382, 383	470, 471, 475, 476
<code>\glsxtrtitlefirst</code>	<code>\glsxtrUpEta</code>
366–368, 382	470, 471, 475, 476
<code>\GLSxtrtitlefirstplural</code>	<code>\glsxtrUpGamma</code>
384	470, 471, 475, 476
<code>\Glsxtrtitlefirstplural</code>	<code>\glsxtrUpIota</code>
366–368, 383	470, 471, 475, 476
<code>\glsxtrtitlefirstplural</code>	<code>\glsxtrUpKappa</code>
366–368, 383	470, 471, 475, 476
<code>\GLSxtrtitlefull</code>	<code>\glsxtrUpLambda</code>
386	470, 471, 475, 476
<code>\Glsxtrtitlefull</code>	<code>\glsxtrUpMu</code>
367, 368, 386	470, 471, 475, 477
<code>\glsxtrtitlefull</code>	<code>\glsxtrUpNu</code>
366–368, 386	470, 471, 475, 477
<code>\GLSxtrtitlefullpl</code>	<code>\glsxtrUpOmega</code>
387	470, 471, 476, 477
<code>\Glsxtrtitlefullpl</code>	<code>\glsxtrUpOmicron</code>
367, 368, 387	470, 471, 476, 477
<code>\glsxtrtitlefullpl</code>	<code>\glsxtrUpPhi</code>
366–368, 387	470, 471, 476, 477
<code>\GLSxtrtitlelong</code>	<code>\glsxtrUpPi</code>
384, 385	470, 471, 476, 477
<code>\Glsxtrtitlelong</code>	<code>\glsxtrUpPsi</code>
366–368, 384	470, 471, 476, 477
<code>\glsxtrtitlelong</code>	<code>\glsxtrUpRho</code>
366–368, 384	470, 471, 476, 477
<code>\GLSxtrtitlelongpl</code>	<code>\glsxtrUpSigma</code>
385	470, 471, 476, 477
<code>\Glsxtrtitlelongpl</code>	<code>\glsxtrUpTau</code>
366–368, 385	470, 471, 476, 477
<code>\glsxtrtitlelongpl</code>	<code>\glsxtrUpTheta</code>
366–368, 385	470, 471, 475, 476
<code>\GLSxtrtitlename</code>	<code>\glsxtrUpUpsilon</code>
380	470, 471, 476, 477
<code>\Glsxtrtitlename</code>	<code>\glsxtrUpXi</code>
366, 367, 380	470, 471, 476, 477
<code>\glsxtrtitlename</code>	<code>\glsxtrUpZeta</code>
366, 367, 380	470, 471, 475, 476
<code>\glsxtrtitleorpdforheading</code>	<code>\GlsXtrUseAbbrStyleFmts</code>
..... 29, 30, 163, 164, 366, 367 255, 257, 261, 264, 266, 268, 269,
<code>\GLSxtrtitleplural</code>	271, 272, 275, 277, 280, 286, 289, 291,
382	293, 296, 303, 306, 308, 310, 312, 314,
<code>\Glsxtrtitleplural</code>	317, 322, 325, 328, 330, 337, 338, 341,
366, 367, 382	343, 346, 347, 349, 353, 355, 358, 361, 364
<code>\glsxtrtitleplural</code>	<code>\GlsXtrUseAbbrStyleSetup</code>
366, 367, 381 266, 268, 269, 271,
<code>\Glsxtrtitleshort</code>	272, 280, 282, 296, 299, 317, 321, 322, 325
366, 367, 379	<code>\glsxtrusefield</code>
<code>\glsxtrtitleshort</code>	439
366, 367, 378	<code>\glsxtruserfield</code>
<code>\Glsxtrtitleshortpl</code>	331
366, 367, 379	<code>\glsxtruserparen</code>
<code>\glsxtrtitleshortpl</code>	332–343
366, 367, 379	<code>\glsxtrusersuffix</code>
<code>\GLSxtrtitletext</code>	332, 334, 339, 342
381	<code>\glsxtruseseealsoformat</code>
<code>\Glsxtrtitletext</code>	60, 62
366, 367, 381	<code>\glsxtruseseeformat</code>
<code>\glsxtrtitletext</code>	58, 61
366, 367, 380, 381	<code>\GlsXtrWarnDeprecatedAbbrStyle</code>
<code>\GlsXtrTotalRecordCount</code>	231, 252
177	<code>\GlsXtrWarning</code>
<code>\glsxtrtreechildpredesc</code>	71, 72
504, 508	<code>\glsxtrword</code>
<code>\glsxtrtreepredesc</code>	230
503, 508	<code>\glsxtrwordsep</code>
<code>\glsxtrtreetopindent</code> 230, 344, 347, 349, 350, 353, 355, 356
508, 517	<code>\glsxtrwrglossmark</code>
<code>\glsxtrundefaction</code>	27
..... 6, 14–16, 34, 53, 55–57, 397, 398, 408	
<code>\glsxtrundeftag</code>	
33, 148, 149	
<code>\GlsXtrUnknownDialectWarning</code>	
49	

H	
<code>\hangindent</code>	504, 506, 508, 518–520, 524–526, 557, 559, 562
<code>\hbox</code>	488
<code>\hfill</code>	488
<code>\href</code>	108
<code>\hsize</code>	73, 74
<code>\hss</code>	488
<code>\hyperlink</code>	16, 109, 442
<code>\hyperpage</code>	216
<code>\hyperref</code>	108, 153, 440, 444
<code>hyperref</code> package	109, 216, 364, 378, 442
I	
<code>\if</code>	70
<code>\if@display</code>	11
<code>\if@glxtr@autoseeindex</code>	30, 31, 57, 64
<code>\if@glxtr@equations</code>	8, 83
<code>\if@glxtr@floats</code>	18
<code>\if@glxtr@format@override</code>	216
<code>\if@glxtr@docdefrestricted</code>	68
<code>\if@glxtr@indexcrossrefs</code>	17, 65
<code>\if@mgl@writesepraterefs</code>	425
<code>\if@org@KV@glslink@local</code>	102
<code>\ifblank</code>	34, 35, 71, 72, 139, 452
<code>\ifbool</code>	33, 51, 397, 406
<code>\ifcase</code> ...	6, 15, 22, 24, 27, 28, 69, 80, 145, 403, 404, 411–413, 415, 416, 419, 502, 530
<code>\ifcsdef</code>	34, 44, 50, 53–55, 80, 84, 100, 107, 114–118, 128, 143, 150, 151, 164, 171, 174, 176, 182, 208, 209, 211–214, 221, 225, 231, 246, 250, 391–393, 420, 421, 442, 492–499, 560
<code>\ifcseempty</code>	400
<code>\ifcsname</code>	490
<code>\ifcsstring</code>	34, 203, 250
<code>\ifcsundef</code>	33, 43, 48, 50, 53, 55, 68, 73, 75, 109, 122, 128–132, 149, 150, 154, 171, 182, 183, 196, 203, 249, 251, 252, 395, 399, 400, 408, 488, 509, 510, 518, 532, 560
<code>\ifcsvoid</code>	65, 103, 202
<code>\ifdef</code> .	15, 22, 30, 33, 36, 43, 44, 47, 49, 51, 56, 57, 62, 63, 67, 73, 74, 102, 106, 107, 115–117, 139, 142, 143, 147, 148, 159, 161, 165, 174, 192, 205–207, 219, 220, 223, 331, 366, 378–387, 425, 426, 434, 440, 445–447, 453, 485, 486, 488–492, 500–503, 505, 507, 519–525, 528, 532, 533
<code>\ifdefempty</code>	6– 8, 41, 43, 48, 49, 53–55, 58, 60, 61, 83, 85, 122, 133, 134, 136, 139, 142, 152, 159, 166, 169, 172, 177, 178, 193–195, 222, 230, 247, 396, 420, 421, 427, 448, 529
<code>\ifdefequal</code>	46, 47, 68, 106, 134, 136, 158, 172, 174, 214, 402
<code>\ifdefstring</code> 5, 14, 26, 46, 138, 162, 216, 222, 528
<code>\ifdefvoid</code>	57, 64–66, 108, 128, 149, 153, 173, 394, 395, 407, 409, 444
<code>\ifdim</code> .	73, 74, 138, 423, 424, 509–517, 537, 561
<code>\IfFileExists</code>	24, 154, 157, 159, 161, 486, 487
<code>\ifglossaryexists</code>	57, 168, 438
<code>\ifgl@sacronym</code>	20, 157, 438
<code>\ifgl@sacrshortcuts</code>	22
<code>\ifgl@automake</code>	142, 157, 161
<code>\ifgl@sentrycounter</code>	49, 50
<code>\ifgl@sentryexists</code>	8, 33, 56, 70–72, 74, 75, 86, 172, 203, 224, 449
<code>\ifgl@sfieldeq</code>	200
<code>\ifgl@shasdesc</code> ...	503, 504, 557, 558, 560, 562
<code>\ifgl@shasfield</code>	36, 37, 331
<code>\ifgl@shaslong</code>	127, 181, 430
<code>\ifgl@shasparent</code> 163, 164, 166, 169, 510, 512, 513
<code>\ifgl@shashshort</code>	58–60, 77, 86, 430
<code>\ifgl@shassymbol</code> 226, 432, 503, 504, 506, 508, 558, 560
<code>\ifgl@indexonlyfirst</code>	104
<code>\ifGlsLongExtraUseTabular</code>	538, 540, 541, 543, 544, 546, 547, 549, 550, 552, 553, 555
<code>\ifgl@nogroupskip</code>	490, 493– 500, 502, 505, 506, 519, 528, 539, 541– 543, 545, 547, 548, 550, 551, 553, 554, 556
<code>\ifgl@nonumberlist</code>	77
<code>\ifgl@nopostdot</code>	18, 19, 144
<code>\ifgl@sanitizesort</code>	141
<code>\ifgl@subentrycounter</code>	49, 50
<code>\ifgl@sused</code>	66, 101, 123, 132, 137, 247, 510–512, 514–516
<code>\ifgl@xindy</code>	154–156
<code>\ifgl@xtr@hyperoutside</code>	84
<code>\ifgl@xtr@printgloss@groups</code> ...	166, 169
<code>\ifgl@xtr@trinitwrglossbefore</code> ...	80, 84, 178
<code>\ifgl@xtr@insertinside</code>	239–246, 249, 254, 256, 257, 259, 260, 262, 263, 265, 267–271, 274–285, 287, 288, 290– 302, 304, 305, 307, 309, 311, 313–327, 329, 332–334, 336, 339, 340, 342–344, 347, 350, 352, 353, 356–358, 360, 362, 363

<code>\ifhaslong</code>	423, 424	J	
<code>\ifhasshort</code>	422–424	<code>\jobname</code>	67, 154–158, 160, 161
<code>\ifHy@hyperindex</code>	216	K	
<code>\ifinlist</code>	119	<code>\key@ifundefined</code>	
<code>\ifinlistcs</code>	38, 68	12, 13, 34, 35, 99, 166, 169, 172
<code>\ifinner</code>	29	<code>\KV@glslink@hyperfalse</code>	87, 101, 102, 109, 110
<code>\ifKV@glslink@hyper</code>	78, 82, 84, 85	<code>\KV@glslink@hypertrue</code>	109
<code>\ifKV@glslink@local</code>	102	<code>\KV@glslink@noindexfalse</code>	86, 102, 103
<code>\ifKV@glslink@noindex</code>	7, 8, 12, 36, 103, 104	<code>\KV@glslink@noindextrue</code>	103, 110
<code>\ifKV@glstrcombined@firstskipmain</code>		<code>\KV@glstrcombined@firstskipmainfalse</code>	
.....	405, 410	390
<code>\ifKV@glstrcombined@firstskipothers</code>		<code>\KV@glstrcombined@firstskipothersfalse</code>	
.....	405, 410	390
<code>\ifKV@glstrcombined@usedskipmain</code> .		<code>\KV@glstrcombined@usedskipmainfalse</code>	
.....	405, 410	391
<code>\ifKV@glstrcombined@usedskipothers</code>		<code>\KV@glstrcombined@usedskipothersfalse</code>	
.....	405, 410	391
<code>\ifKV@mgl@presetlocal</code>	403	<code>\KV@mgl@presetlocalfalse</code>	401
<code>\ifmglused</code>	408, 409		
<code>\ifmmode</code>	11, 29, 30	L	
<code>\ifmultiglossaryentryglobal</code>	394–396, 399	<code>\L</code>	467, 470
<code>\ifnum</code>	16, 17, 43, 67, 124,	<code>\l</code>	470
132, 133, 149, 160, 172, 177, 392, 393,		<code>\label</code>	26, 146
395, 396, 399, 401, 406, 410, 445–447,		<code>\large</code>	559
504, 506, 518, 519, 529, 530, 557, 561, 562		<code>\LaTeX</code>	155, 156
<code>\ifst@rred</code>	11	<code>\leaders</code>	488
<code>\ifstrempty</code>		<code>\leavevmode</code>	52, 82
164, 174, 183, 409, 442, 443, 445, 446, 448		<code>\let</code>	4, 6–
<code>\ifstrequal</code>	19, 24, 82, 106, 442	10, 12, 14–16, 18, 20–22, 27, 28, 30–32,	
<code>\ifthenelse</code>	157, 158, 225	36, 39, 47, 48, 52, 61, 67–69, 73, 75, 76,	
<code>\IfTrackedDialectHasMapping</code>	48	78, 79, 82–98, 100–106, 109, 110, 113,	
<code>\IfTrackedLanguageFileExists</code>	437	119–123, 130–140, 142–150, 159, 161,	
<code>\ifundef</code>	16, 26,	162, 165–170, 172, 174, 177, 178, 183,	
41, 43, 48, 139, 222, 223, 393, 442, 485, 537		192, 194, 208–218, 221–223, 228, 230–	
<code>\ifx</code>	7,	232, 236–246, 249, 251, 262, 264, 287,	
9–11, 14, 32, 61, 63, 73, 74, 83, 136, 139,		289, 304, 306, 328, 330, 365–368, 390,	
143, 146, 151–153, 159, 161, 162, 167,		392–397, 399, 401, 402, 405–415, 417–	
168, 216–218, 220, 228–230, 232, 344,		420, 425, 438, 440, 443, 451–453, 485,	
394, 399, 402, 404, 405, 407, 409, 411,		486, 489, 490, 501, 508, 510, 521, 529–532	
413–415, 417, 418, 425, 426, 444, 452, 526		<code>\letabbreviationstyle</code>	
<code>\immediate</code> 67, 123, 124, 132, 154, 161, 425, 426		260, 261, 263, 264, 266,
<code>\index</code>	217	268, 271, 272, 278, 280, 295, 296, 315, 317	
<code>\indexspace</code>	489, 501, 528, 532	<code>\letcs</code>	34, 41, 43, 58,
<code>\input</code>	437	60, 61, 66, 80, 84, 99, 107, 108, 147–150,	
<code>\inputencodingname</code>	161	171, 172, 208–215, 221, 399, 409, 452, 533	
<code>\InputIfFileExists</code>	67	<code>\levelchar</code>	219
<code>\istfilename</code>	139	<code>\linewidth</code>	537
<code>\item</code>	156, 488, 489, 491, 501, 502, 520, 521	<code>\listadd</code>	128
		<code>\listbreak</code>	222

<code>\listcsadd</code>	37	<code>\mglscurrentmainlabel</code>	
<code>\listcseadd</code>	37, 129	403–405, 409, 410, 414, 415, 417, 452
<code>\listcsgadd</code>	37, 68	<code>\mglscurrentmainlist</code>	452
<code>\listcsxadd</code>	37, 129	<code>\mglscurrentmainoptions</code>	452
<code>\listxadd</code>	119, 393	<code>\mglscurrentmultilabel</code> .	408, 409, 411, 419
<code>\loadglsentries</code>	69, 155	<code>\mglscurrentoptions</code>	409
<code>\long</code>	52	<code>\mglscurrentprefix</code>	411, 420
M			
<code>\m@ne</code>	439	<code>\mglscurrentsuffix</code>	411, 421
<code>\MakeAcronymsAbbreviations</code>	137	<code>\mglscustompostlinkhook</code>	412
<code>\makeatletter</code>	67, 154, 159, 218	<code>\mglselementindex</code> 394, 395, 399, 400, 406, 413	
<code>\makeatother</code>	218	<code>\mglselementposthook</code>	418
<code>\makebox</code>	488, 518, 519, 561	<code>\mglselementprehook</code>	415
<code>\makefirstuc</code>	223, 453	<code>\mglselementreset</code>	403, 404
<code>makeglossaries</code>	147	<code>\mglselementunset</code>	404, 405
<code>\makeglossaries</code>	138, 154, 156–158, 162	<code>\mgl\$field</code>	433
<code>makeindex</code>	15, 138	<code>\mgl\$forotherelements</code>	402
<code>\makenoidxglossaries</code>	156	<code>\mgl\$hascategoryprefix</code>	420
<code>\MakeTextUppercase</code>	367	<code>\mgl\$hascategorysuffix</code>	421
<code>\MakeUppercase</code>	366, 367	<code>\mgl\$iflast</code>	406, 413, 414, 417
<code>\marginpar</code>	29	<code>\mgl\$iflastelementcaps</code>	
<code>\markboth</code>	365	408, 411, 419, 420
<code>\markright</code>	365	<code>\mgl\$iflastelementsskipped</code> ..	408, 411, 418
<code>\mathit</code>	455	<code>\mgl\$iflastelementwasfirstuse</code>	
<code>\mathrm</code>	454–456	408, 411, 419, 420
<code>\maxdimen</code>	73, 74	<code>\mgl\$iflastelementwasplural</code>	
<code>\mbox</code>	490, 491, 518	408, 411, 419, 420
<code>\medskip</code>	157, 158, 171, 560	<code>\mgl\$iflastmaincaps</code>	408, 418, 420
<code>\MessageBreak</code>	69,	<code>\mgl\$iflastmainsskipped</code>	408, 418
	72, 124, 133, 139, 142, 143, 158, 167, 250	<code>\mgl\$iflastmainwasfirstuse</code> .	408, 418, 420
<code>mfistuc package</code>	222	<code>\mgl\$iflastmainwasplural</code> ...	408, 418, 420
<code>\mfistucMakeUppercase</code>	44, 87–98,	<code>\mgl\$isfirstuse</code> .	405, 408–411, 413, 415, 416
	101, 111–113, 115, 117, 118, 126, 127,	<code>\mgl\$lastcategory</code>	408, 411
	135, 181, 184–190, 197–200, 210, 211,	<code>\mgl\$lastelementlabel</code> 408, 411, 413, 414, 420	
	215, 237, 239, 241, 242, 244, 246–249, 453	<code>\mgl\$lastelementpostlinkhook</code> ..	411, 412
<code>\mfu@checkword@arg</code>	222, 223	<code>\mgl\$lastmainlabel</code>	408, 417, 418, 420
<code>\mfu@checkword@do</code>	223	<code>\mgl\$lastmainpostlinkhook</code>	411, 412
<code>\mgl\$</code>	425, 426	<code>\mgl\$lastmultilabel</code>	408, 411
<code>\mgl\$@disable@mglsopts</code>	407	<code>\mgl\$localunset</code>	419
<code>\mgl\$@disable@setup</code>	409	<code>\mgl\$name</code>	62
<code>\mgl\$@do@postlinkhook</code> ..	408, 411, 412, 419	<code>\mgl\$prefix</code>	413
<code>\mgl\$@enable@mglsopts</code>	407	<code>\mgl\$reset</code>	399
<code>\mgl\$@enable@setup</code>	409, 410	<code>\mgl\$seefirstitem</code>	62
<code>\mgl\$@org@postlinkhook</code>	413, 414	<code>\mgl\$seeitem</code>	62
<code>\mglscurrentcategory</code> ...	410, 411, 420, 421	<code>\mgl\$suffix</code>	419
<code>\mglscurrentlabel</code>		<code>\mgl\$unset</code>	398, 419
.....	405, 413–415, 417, 418, 452, 453	<code>\mgl\$usecategoryprefix</code>	420
<code>\mglscurrentlist</code>	403–405, 409, 413	<code>\mgl\$usecategorysuffix</code>	421
		<code>\mgl\$wasfirstuse</code>	408, 411, 412
		<code>\mgl\$WriteSeparateRefsFalse</code>	426

<code>\mglWriteSeparateRefsTrue</code>	425	<code>\newglossary</code>	20, 139, 140
<code>\midrule</code>	536, 539, 541, 542, 544, 545, 547, 549, 550, 552, 553, 555	<code>\newglossaryentry</code>
<code>\mpgls</code>	434	...	22, 67, 69, 122, 130, 134, 205, 206, 233
<code>\mpglsWarning</code>	434	<code>\newglossaryentry options</code>	
<code>\multiglossaryentry</code>	397, 398, 408	access	194
<code>\multiglossaryentryglobalfalse</code>	392	alias	17, 57, 61, 63–66
<code>\multiglossaryentryglobaltrue</code>	397	desc	187, 188, 198, 199
N			
<code>\NeedsTeXFormat</code> ...	4, 438, 487, 527, 534, 557	descplural	188, 199
<code>\new@atom@glossaryentry</code>	67	first	107, 185, 198, 253, 373, 374, 382
<code>\new@command</code>	440	firstaccess	195
<code>\new@glossaryentry</code>	69, 142	firstplural	186, 198, 253, 373, 374, 383
<code>\new@ifnextchar</code> ..	36, 100, 101, 126, 127, 175, 178–180, 227, 236–246, 426, 449, 450	group	171, 172
<code>\newabbr</code>	21	loclist	37
<code>\newabbreviation</code>	21	long	190, 200, 384
<code>\newabbreviationhook</code>	233	longplural	190, 200, 385
<code>\newabbreviationstyle</code>	name ...	58, 183, 184, 197, 216, 370, 371, 379
...	253, 255, 257, 258, 260, 261, 263, 264, 266, 268, 269, 271–278, 280, 282, 284, 286, 288, 290, 291, 293, 295– 297, 299, 300, 302, 303, 305–315, 317– 319, 321–323, 325, 327, 328, 330, 332, 333, 335, 337, 338, 340–343, 345–347, 349, 350, 352–354, 356, 358, 359, 361–363	plural	184, 185, 197, 253, 372, 381
<code>\newacronym</code>	134, 136	see	17, 31, 57, 63, 66, 69, 140
<code>\newacronymhook</code>	134	seealso	17, 57, 60, 63, 65, 66
<code>\newacronymstyle</code>	135, 137	short	58, 189, 199, 229
<code>\newboolean</code>	396	shortaccess	193
<code>\newcommand</code>	4–14, 16–27, 29–31, 33– 47, 49–62, 64, 66, 67, 69–73, 75–77, 80– 82, 85–87, 94, 99–101, 103–105, 107– 110, 113, 115–122, 124, 126–130, 132, 133, 135–138, 143–150, 152, 153, 155– 160, 162–168, 170–172, 174–193, 196– 207, 213, 214, 216–231, 233–247, 249– 255, 257, 258, 260, 264, 266, 269, 271– 273, 289, 306, 331, 334–337, 340, 344, 345, 347, 349, 350, 353, 355, 356, 359, 361–363, 365–394, 396–408, 420–423, 425–427, 433, 434, 437, 439–445, 447– 450, 452, 453, 456–485, 487, 489–491, 500, 501, 503–510, 517, 518, 527–529, 532–542, 544, 545, 547–553, 555, 558–561	shortaccessplural	194
<code>\newcount</code>	146, 160, 174, 392	shortplural	190, 199, 229
<code>\newcounter</code>	26, 182	symbol	186, 187, 198, 432
<code>\newentry</code>	22	symbolplural	187, 198
<code>\newenvironment</code>	167	text ...	58, 107, 184, 197, 253, 255, 371, 380
		textaccess	195
		user2	49
		<code>\newglossarystyle</code>
		...	529, 538, 540, 541, 543, 544, 546, 547, 549, 550, 552, 553, 555, 557, 561
		<code>\newif</code>	80, 215, 253, 392, 396, 425, 538
		<code>\newlength</code>	508, 559
		<code>\newnum</code>	22
		<code>\newrobustcmd</code>
		...	35, 37–39, 41, 45–47, 61, 63, 68, 81, 86, 100, 101, 113–115, 126, 144, 150, 163, 164, 170, 171, 174–176, 178–180, 214, 221–223, 236–247, 344, 366, 368–378, 392, 393, 399, 400, 426, 449–452, 510–517
		<code>\newsym</code>	22
		<code>\newterm</code>	205
		<code>\newtoks</code>	229
		<code>\newwrite</code>	67, 139
		<code>\nfss@text</code>	29, 30
		<code>\nobreak</code>	489–491, 501, 558
		<code>\NoCaseChange</code>	115–118, 164, 368–378
		<code>\noexpand</code>	10, 12, 24, 61, 64, 65, 67, 82, 134, 154, 159, 167,

169, 170, 172, 182, 193, 217, 218, 233, 407, 409–411, 414, 417, 419, 426, 439, 449, 487, 529–531, 538–552, 554, 555, 562	true 161
\nofiles 157	autoseeindex 30
\noindent ... 157, 158, 505, 507, 521–525, 558	false 30
\nopagebreak 501, 528, 532, 557, 562	counter
\nopostdesc 52, 71, 72, 144, 168, 205	wrglossary 26
\np@glstr@assign@leveloffset 146	debug
\ns@GLSxtrfull 237	showtargets 27, 109
\ns@GLSxtrfull 236	docdef 16, 68, 69, 122, 130, 396
\ns@glstrfull 236	atom 67
\ns@GLSxtrfullpl 238	false 68, 69
\ns@GLSxtrfullpl 238	restricted 17
\ns@glstrfullpl 237	true 67, 69
\ns@GLSxtrlong 242	docdefs
\ns@GLSxtrlong 241, 242	restricted 68
\ns@glstrlong 241	equations 8, 83
\ns@GLSxtrlongpl 246	indexcounter 440
\ns@GLSxtrlongpl 245	nonumberlist 74
\ns@glstrlongpl 245	nopostdot 18
\ns@GLSxtrshort 240	false 18
\ns@GLSxtrshort 240	numbers 21
\ns@glstrshort 239	order
\ns@GLSxtrshortpl 244	letter 162
\ns@GLSxtrshortpl 243	postdot 18
\ns@glstrshortpl 243	prefix 23
\null 24	record 6,
\number 67, 129, 130, 132, 159, 173, 174, 395, 400	13, 15, 68, 78, 138, 158, 159, 236, 238–246
\numexpr 129, 132, 172, 395, 400	alsoindex 7, 9, 13, 15, 162
	hybrid 9, 13, 15
O	nameref 11, 32, 159, 162
\O 467, 470	only 7, 157
\o 470	shortcuts 22
\openout 67	ac 22
\or 6, 15, 16, 22, 23, 27,	all 22
28, 69, 80, 403–405, 411–416, 419, 502, 531	false 22
\org@glossaryentrynumbers ... 74, 143, 168	none 22
\org@glossarytitle 143, 167, 168	true 22
\org@ifKV@glslink@hyper 82, 85	sort
	use 85
P	style 25
\p@glshyp@opt 105	stylemods 25
\p@glstr@assign@leveloffset 146	symbols 21, 206
\p@GLSxtrMglsOrGls 427	undefaction 56
package options:	error 5
abbreviations 20	warn 5
accsupp 23, 183	xindy 62, 63
acronym 20	\PackageError 5, 12, 25, 30, 67, 69,
automake 142, 155, 161	72, 73, 81, 100, 101, 104, 106, 114, 122,
immediate 161	123, 130, 132, 133, 137, 139, 141, 142,

<code>\renewenvironment</code>	490, 492–499, 501, 504, 506, 518, 521–525, 529, 538–544, 546–552, 554, 555, 557, 561
<code>\renewglossarystyle</code>	488, 490–499, 501, 502, 504–507, 518–525
<code>\renewrobustcmd</code>	61, 85, 109
<code>\RequireGlossariesExtraLang</code> ...	437, 486
<code>\RequirePackage</code>	4, 15, 23–25, 28, 487, 527, 534, 557
<code>\reserved@a</code>	228
<code>\reserved@b</code>	228
<code>\reserved@d</code>	228
<code>\RestoreAcronyms</code>	134, 137
<code>\rGLS</code>	177
<code>\rGls</code>	176
<code>\rgls</code>	176
<code>\rGLSformat</code>	180
<code>\rGlsformat</code>	179
<code>\rglsformat</code>	179, 181
<code>\rGLSpl</code>	177
<code>\rGlspl</code>	177
<code>\rglspl</code>	177
<code>\rGLSplformat</code>	180
<code>\rGlsplformat</code>	180
<code>\rglsplformat</code>	179, 181
<code>\robustify</code>	439
<code>\romannumeral</code>	509, 510, 518, 560
S	
<code>\s@glsxtrfmt</code>	36
<code>\s@gls@hyp@opt</code>	105
<code>\s@glsxtr@enabletagging</code>	222
<code>\s@glsxtrfmt</code>	35
<code>\s@glsxtrforcsvfield</code>	38
<code>\s@GlsXtrIfFieldCmpNum</code>	42
<code>\s@GlsXtrIfFieldEqNum</code>	42
<code>\s@GlsXtrIfFieldEqStr</code>	46
<code>\s@GlsXtrIfFieldEqXpStr</code>	46
<code>\s@GlsXtrIfFieldNonZero</code>	42, 439
<code>\s@GlsXtrIfFieldValueInCsvList</code>	39
<code>\s@glsxtrifhasfield</code>	39–41, 46, 47
<code>\s@GlsXtrIfHasNonZeroChildCount</code> ...	439
<code>\s@GlsXtrIfValueInFieldCsvList</code>	40
<code>\s@GlsXtrIfXpFieldEqXpStr</code>	47
<code>\s@GlsXtrMglsOrGls</code>	427
<code>\s@GlsXtrStartUnsetBuffering</code>	119
<code>\s@GlsXtrStopUnsetBuffering</code>	119
<code>\s@ifglossaryexists</code>	33
<code>\s@printunsrtglossary</code>	165, 170
<code>\s@xGlsXtrIfValueInFieldCsvList</code>	40
<code>\seealsoname</code>	61, 63
<code>\seename</code>	58, 61
<code>\setabbreviationstyle</code>	137, 254, 266
<code>\SetAcronymLists</code>	135–137
<code>\setacronymstyle</code>	135, 137
<code>\SetDefaultAcronymDisplayStyle</code>	137
<code>\setentrycounter</code>	151, 442, 443
<code>\SetGenericNewAcronym</code>	137
<code>\setglossarystyle</code>	25, 143, 167, 488, 490–492, 502, 505, 507, 519–526, 529
<code>\setkeys</code>	8, 25, 31, 36, 83, 85, 104, 134, 143, 167, 168, 178, 231, 232, 388, 407, 409, 410
<code>\setlength</code>	73, 74, 501, 504, 506, 508, 519, 521, 522, 524, 529, 537, 538, 559, 560
<code>\settowidth</code> .	138, 423, 424, 508–518, 537, 560
<code>\setupglossaries</code>	4, 31
<code>\sfcode</code>	18, 19, 225, 500
<code>\small</code>	60
<code>\smallskip</code>	560
<code>soul package</code>	119
<code>\space</code>	5, 12, 14, 62, 69, 70, 72, 94, 104, 122–124, 130, 132– 134, 137–141, 143, 154, 157, 158, 162, 167, 170, 171, 174, 226, 230, 234, 254, 395, 421, 423–426, 434, 488, 490, 500, 503, 504, 506, 508, 518, 527, 558, 560, 561
<code>\spacefactor</code>	18, 19, 225, 500
<code>\stepcounter</code>	182
<code>\string</code>	5, 10–12, 14, 32, 62, 63, 68–70, 72, 76, 84, 94, 100, 101, 104, 106, 107, 114, 115, 122–124, 130, 132– 134, 137–143, 154–163, 167, 170, 171, 174, 210–213, 215, 217, 225, 397, 398, 408, 425, 426, 434, 439, 449, 456–485, 532
<code>\strut</code>	488, 490–500, 506, 535
<code>\subglossentry</code>	144, 168, 173, 488, 490–500, 502, 504, 506, 518, 530, 539, 541–543, 545, 546, 548, 550, 551, 553, 554, 556, 557, 562
<code>\subitem</code>	501, 502
<code>\subsubitem</code>	501, 502
<code>\symbolname</code>	537, 544, 545, 547, 548, 550, 552, 553, 555
T	
<code>\tabcolsep</code>	537, 538
<code>\tablehead</code>	496–499
<code>\tabletail</code>	496–499

