

glossaries-extra.sty v1.46: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2021-09-20

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	32
1.3	Modifications to Commands Provided by glossaries	48
1.3.1	Existence Checks	53
1.3.2	Document Definitions	63
1.3.3	Existing Glossary Style Modifications	69
1.3.4	Entry Formatting, Hyperlinks and Indexing	73
1.3.5	Entry Counting	113
1.3.6	Acronym Modifications	128
1.3.7	Indexing and Displaying Glossaries	132
1.4	Link Counting	176
1.5	Integration with glossaries-accsupp	178
1.6	Categories	195
1.7	Abbreviations	222
1.7.1	Abbreviation Styles Setup	243
1.7.2	Predefined Styles (Default Font)	246
1.7.3	Predefined Styles (Small Capitals)	266
1.7.4	Predefined Styles (Fake Small Capitals)	282
1.7.5	Predefined Styles (Emphasized)	299
1.7.6	Predefined Styles (User Parentheses Hook)	324
1.7.7	Predefined Styles (Hyphen)	334
1.7.8	Predefined Styles (No Short on First Use)	349
1.8	Using Entries in Headings	352
1.9	Multi-Lingual Support	375
1.10	glossaries-extra-bib2gls.sty	376
2	Style Adjustments (glossaries-extra-stylemods.sty)	422
2.1	Package Initialisation	422
2.2	List-Like Styles	423
2.3	Longtable Styles	426
2.4	Long Ragged Styles	428
2.5	Supertabular Styles	430
2.6	Super Ragged Styles	432
2.7	Inline Style	434
2.8	Tree Styles	434
2.9	Multicolumn Styles	454

3	bookindex style (glossary-bookindex.sty)	461
3.1	Package Initialisation and Options	461
4	longextra styles (glossary-longextra.sty)	468
4.1	Package Initialisation and Options	468
5	topic styles (glossary-topic.sty)	491
5.1	Package Initialisation and Options	491
	Glossary	497
	Change History	498
	Index	525

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2021/09/20 v1.46 (NLCT)]

Requires xkeyval to define package options.
3 \RequirePackage{xkeyval}

Requires etoolbox package.
4 \RequirePackage{etoolbox}

Has glossaries already been loaded?
5 \@ifpackageloaded{glossaries}
6 {%

Already loaded so pass any options to \setupglossaries. This means that the options that
can only be set when glossaries is loaded can't be used.
7   \newcommand{\glstr@dooption}[1]{\setupglossaries{#1}}%
8   \let\@glstr@declareoption\@gls@declareoption
9 }
10 {%

Not already loaded, so pass options to glossaries.
11   \newcommand{\glstr@dooption}[1]{%
12     \PassOptionsToPackage{#1}{glossaries}%
13   }%

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

Declare package options.

`\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\glstr@undefaction\glstr@warn@undefaction
69      \let\glstr@warnonexistsordo\glstr@warn@onexistsordo
70      \let\glstr@redef@for\glstr@sentries\glstr@do@redef@for\glstr@sentries
71    \or
72      \let\glstr@undefaction\glstr@err@undefaction
73      \let\glstr@warnonexistsordo\glstr@gobble
74      \let\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 \ifglentryexists{#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\theglsentrycounter
175 \let\@glsxtr@orgprefix\@glo@counterprefix
176 \ifx\theglsentrycounter\theHglentrycounter
177   \def\@glo@counterprefix{}%
178 \else
  Protect against non-expandable commands occurring in the location.
179   \protected@edef\@glsxtr@theentrycounter{\theglsentrycounter}%
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 \theglsentrycounter 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\theglsentrycounter\theHglentrycounter
202     \ifx\@glsxtr@record@setting\@glsxtr@record@setting@nameref
203       \@glsxtr@do@nameref@record
204       {\@gls@label}{\@gls@counter}{\@glsnumberformat}%
205       {\theglsentrycounter}%
206     \else
207       \protected@write\@auxout{}{\string\@glsxtr@record
208         {\@gls@label}{\@gls@counter}{\@glsnumberformat}%
209         {\theglsentrycounter}}%
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   \@glxtrwrglossmark
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@warnonexistssordo\glxtr@warn@onexistssordo
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
399 % \end{macrocode}
400 % Hybrid record (use bib2gls to fetch definitions) and index (use
401 % makeindex/xindy to sort and collate).
402 % \begin{macrocode}
403 \def\glxtr@setup@record{%
404 \renewcommand*{\@glxtr@record@setting@alsoindex}{hybrid}%
405 \renewcommand*{\@do@seeglossary}{\@glxtr@dosee@alsoindex@glossary}%
406 \let\@glxtr@record\@glxtr@record
407 \let\@do@wrglossary\glxtr@do@alsoindex@wrglossary
408 \let\@glxtr@saveentrycounter\glxtr@indexonly@saveentrycounter
409 \let\glxtrundefaction\glxtr@warn@undefaction
410 \let\glxtr@warnonexistssordo\glxtr@warn@onexistssordo
411 \glxtr@addloclistfield
412 \let\@glxtr@recordcounter\glxtr@op@recordcounter
413 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
414 \undef\glxtrsetaliasnoindex
415 }%
416 \fi
417 }

```

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).

```

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

```

Need to provide conditional commands that are backward compatible:

`\if@glxtrdocdef`

```

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

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

sxtrdocdeffalse
421 \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.
422 \define@choicekey{glossaries-extra.sty}{docdef}
423 [\@glxtr@docdefsetting\@glxtr@docdefval]%
424 {false,true,restricted,atom}[true]%
425 {%
426 \ifnum\@glxtr@docdefval>1\relax
427 \renewcommand*{\@glsdofexistsorwarn}{\glsdofexists}%
428 \else
429 \renewcommand*{\@glsdofexistsorwarn}{\glsdofexistsorwarn}%
430 \fi
431 }

ocdefrestricted
432 \newcommand*{\if@glxtrdocdefrestricted}{\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).
433 \newcommand*{\@glsdofexistsorwarn}{\glsdofexistsorwarn}

indexcrossrefs Automatically index cross references at the end of the document
434 \define@boolkey{glossaries-extra.sty}{\@glxtr}{indexcrossrefs}[true]{%
435 \if@glxtrindexcrossrefs
436 \else
437 \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
438 \fi
439 }

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

But allow see key to switch it on automatically.

oindexcrossrefs
441 \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.

```

442 \define@boolkey{glossaries-extra.sty}[@glxtr@]{autoseeindex}[true]{%
443 }
444 \@glxtr@autoseeindextrue

```

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

```

445 \define@boolkey{glossaries-extra.sty}[@glxtr@]{equations}[true]{%
446 }
447 \@glxtr@equationsfalse

```

\glxtr@float

```

448 \let\glxtr@float\@float

```

glxtr@dblfloat

```

449 \let\glxtr@dblfloat\@dblfloat

```

floats Provide a boolean option to automatically switch to the the corresponding counter when in a float.

```

450 \define@boolkey{glossaries-extra.sty}[@glxtr@]{floats}[true]{%
451   \if@glxtr@floats
452     \renewcommand*{\@float}[1]{\renewcommand{\glscounter}{##1}\glxtr@float{##1}}%
453     \renewcommand*{\@dblfloat}[1]{\renewcommand{\glscounter}{##1}\glxtr@dblfloat{##1}}%
454   \else
455     \let\@float\glxtr@float
456     \let\@dblfloat\glxtr@dblfloat
457   \fi
458 }
459 \@glxtr@floatsfalse

```

iesExtraWarning Allow users to suppress warnings.

```

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

```

raWarningNoLine Allow users to suppress warnings.

```

461 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
462   \PackageWarningNoLine{glossaries-extra}{#1}}

463 \@glxtr@declareoption{nowarn}{%
464   \let\GlossariesExtraWarning\@gobble
465   \let\GlossariesExtraWarningNoLine\@gobble
466   \glxtr@doption{nowarn}%
467 }

```

xtr@defpostpunc Redefines \glspostdescription. The postdot and nopostdot options will have to redefine this.

```

468 \newcommand*{\@glxtr@defpostpunc}{%

```

postdot Shortcut for nopostdot=false

```
469 \@glsxtr@declareoption{postdot}{%
470   \glsxtr@doption{nopostdot=false}%
471   \renewcommand*{\@glsxtr@defpostpunc}{%
472     \renewcommand*{\glspostdescription}{%
473       \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
474   }%
475 }
```

nopostdot Needs to redefine \@glsxtr@defpostpunc

```
476 \define@choicekey{glossaries-extra.sty}{nopostdot}{true,false}[true]{%
477   \glsxtr@doption{nopostdot=#1}%
478   \renewcommand*{\@glsxtr@defpostpunc}{%
479     \renewcommand*{\glspostdescription}{%
480       \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
481   }%
482 }
```

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

```
483 \define@key{glossaries-extra.sty}{postpunc}{%
484   \glsxtr@doption{nopostdot=false}%
485   \ifstrequal{#1}{dot}%
486   {%
487     \renewcommand*{\@glsxtr@defpostpunc}{%
488       \renewcommand*{\glspostdescription}{.\spacefactor\sfcode'\. }%
489     }%
490   }%
491   {%
492     \ifstrequal{#1}{comma}%
493     {%
494       \renewcommand*{\@glsxtr@defpostpunc}{%
495         \renewcommand*{\glspostdescription}{,}%
496       }%
497     }%
498     {%
499       \ifstrequal{#1}{none}%
500       {%
501         \glsxtr@doption{nopostdot=true}%
502         \renewcommand*{\@glsxtr@defpostpunc}{%
503           \renewcommand*{\glspostdescription}{}%
504         }%
505       }%
506       {%
507         \renewcommand*{\@glsxtr@defpostpunc}{%
508           \renewcommand*{\glspostdescription}{#1}%
509         }%
510       }%
511     }%
512   }%
513 }
```

```

512 }%
513 }

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

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

abbreviationsdef
516 \newcommand*{\@glsxtr@doabbreviationsdef}{%
517   \ifpackageloaded{babel}%
518   {\providecommand{\abbreviationsname}{\acronymname}}%
519   {\providecommand{\abbreviationsname}{Abbreviations}}%
520   \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
521   \renewcommand*{\glsxtrabbrvtype}{abbreviations}%
522   \newcommand*{\printabbreviations}[1][{}]{%
523     \printglossary[type=glsxtrabbrvtype,##1]%
524   }%
525   \disable@keys{glossaries-extra.sty}{abbreviations}%

   If the acronym option hasn't been used, change \acronymtype to \glsxtrabbrvtype.
526   \ifglsacronym
527   \else
528     \renewcommand*{\acronymtype}{\glsxtrabbrvtype}%
529   \fi
530 }%

abbreviations If abbreviations, create a new glossary type for abbreviations.
531 \@glsxtr@declareoption{abbreviations}{%
532   \let\@glsxtr@abbreviationsdef\@glsxtr@doabbreviationsdef
533 }

AbbreviationShortcuts 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).
534 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
535   \newcommand*{\ab}{\cgl}s}%
536   \newcommand*{\abp}{\cgl spl}%
537   \newcommand*{\as}{\glsxtrshort}%
538   \newcommand*{\asp}{\glsxtrshortpl}%
539   \newcommand*{\al}{\glsxtrlong}%
540   \newcommand*{\alp}{\glsxtrlongpl}%
541   \newcommand*{\af}{\glsxtrfull}%
542   \newcommand*{\afp}{\glsxtrfullpl}%
543   \newcommand*{\Ab}{\cGls}%
544   \newcommand*{\Abp}{\cGls pl}%
545   \newcommand*{\As}{\Glsxtrshort}%
546   \newcommand*{\Asp}{\Glsxtrshortpl}%

```

```

547 \newcommand*{\Al}{\Glsxtrlong}%
548 \newcommand*{\Alp}{\Glsxtrlongpl}%
549 \newcommand*{\Af}{\Glsxtrfull}%
550 \newcommand*{\Afp}{\Glsxtrfullpl}%
551 \newcommand*{\AB}{\cGLS}%
552 \newcommand*{\ABP}{\cGLSpl}%
553 \newcommand*{\AS}{\GLSxtrshort}%
554 \newcommand*{\ASP}{\GLSxtrshortpl}%
555 \newcommand*{\AL}{\GLSxtrlong}%
556 \newcommand*{\ALP}{\GLSxtrlongpl}%
557 \newcommand*{\AF}{\GLSxtrfull}%
558 \newcommand*{\AFP}{\GLSxtrfullpl}%

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

```

Disable this command after it's been used.

```

560 \let\GlsXtrDefineAbbreviationShortcuts\relax
561 }

```

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

```

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

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

```

Disable this command after it's been used.

```

588 \let\GlsXtrDefineAcShortcuts\relax
589 }

```

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

```

590 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
591 \newcommand*{\newentry}{\newglossaryentry}%
592 \ifdef\printsymbols
593 {%
594 \newcommand*{\newsym}{\glstrnewsymbol}%
595 }{}%
596 \ifdef\printnumbers
597 {%
598 \newcommand*{\newnum}{\glstrnewnumber}%
599 }{}%
600 \let\GlsXtrDefineOtherShortcuts\relax
601 }

```

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.

```

602 \newcommand*{\@glstr@setupshortcuts}{}

```

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

```

603 \newcommand*{\@glstr@shortcutsval}{\ifglstrshortcuts 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).

```

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

```

```

620     }%
621 \or % abbr
622   \renewcommand*{\@glxtr@setupshortcuts}{%
623     \GlsXtrDefineAbbreviationShortcuts
624   }%
625 \or % other
626   \renewcommand*{\@glxtr@setupshortcuts}{%
627     \GlsXtrDefineOtherShortcuts
628   }%
629 \or % all
630   \renewcommand*{\@glxtr@setupshortcuts}{%
631     \glacrshortcutstrue
632     \GlsXtrDefineAcShortcuts
633     \GlsXtrDefineAbbreviationShortcuts
634     \GlsXtrDefineOtherShortcuts
635   }%
636 \or % true
637   \renewcommand*{\@glxtr@setupshortcuts}{%
638     \glacrshortcutstrue
639     \GlsXtrDefineAcShortcuts
640     \GlsXtrDefineAbbreviationShortcuts
641     \GlsXtrDefineOtherShortcuts
642   }%
643 \or % ac
644   \renewcommand*{\@glxtr@setupshortcuts}{%
645     \glacrshortcutstrue
646     \GlsXtrDefineAcShortcuts
647   }%
648 \else % none, false
649   \renewcommand*{\@glxtr@setupshortcuts}{}%
650 \fi
651 }

```

Leave none and false as last option.

lsxtr@doaccsupp

```
652 \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.

```

653 \@glxtr@declareoption{accsupp}{%
654 \renewcommand*{\@glxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}

```

tr@doloadprefix

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


prefix If prefix, load glossaries-prefix package.

```
656 \@glxtr@declareoption{prefix}{%
657 \renewcommand*{\@glxtr@doloadprefix}{\RequirePackage{glossaries-prefix}}}
```

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

```
658 \newcommand{\glxtrNoGlossaryWarning}[1]{%
659 \GlossariesExtraWarning{Glossary ‘#1’ is missing}%
660 \@glxtr@defaultnoglossarywarning{#1}%
661 }
```

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

```
662 \define@choicekey{glossaries-extra.sty}{nomissingglstext}
663 [ \@glxtr@nomissingglstextval \@glxtr@nomissingglstextnr ]%
664 {true,false}[true]{%
665 \ifcase \@glxtr@nomissingglstextnr \relax % true
666 \renewcommand{\glxtrNoGlossaryWarning}[1]{\null}%
667 \else % false
668 \renewcommand{\glxtrNoGlossaryWarning}[1]{%
669 \@glxtr@defaultnoglossarywarning{#1}%
670 }%
671 \fi
672 }
```

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

xtr@redefstyles

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

stylemods

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

```

693         \eappto\@glsxtr@redefstyles{%
694             \noexpand\RequirePackage{glossary-\@glsxtr@tmp}}}%
695     }%
696     {%
697         \PackageError{glossaries-extra}%
698             {Glossaries style package ‘glossary-\@glsxtr@tmp.sty’
699             doesn’t exist (did you mean to use the ‘style’ key?)}%
700             {The list of values (#1) in the ‘stylemods’ key should
701             match the glossary-xxx.sty files provided with
702             glossaries.sty}%
703     }%
704 }%
705 \appto\@glsxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}}%
706 }
707 }%
708 }

```

glsxtr@do@style

```

709 \newcommand*{\@glsxtr@do@style}{}

```

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

```

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

```

Defer actual style change:

```

711 \renewcommand*{\@glsxtr@do@style}{%

```

Set this as the default style:

```

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

```

Set this style:

```

713 \setglossarystyle{#1}%

```

```

714 }%

```

```

715 }

```

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.

```

716 \newcommand*{\glsxtr@inc@wrglossaryctr}[1]{}

```

cationHyperlink

`\glsxtrinternallocationhyperlink{<counter>}{<prefix>}{<location>}`

The first two arguments are always control sequences.

```

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

```

```

718 \glsxtrhyperlink{#1#2#3}{#3}%

```

```

719 }

```

cationhyperlink

```
720 \newcommand*{\@glxstr@wrglossary@locationhyperlink}[3]{%
721   \pageref{wrglossary.#3}%
722 }
```

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.

```
723 \@glxstr@declareoption{indexcounter}{%
724   \glxstr@doption{counter=wrglossary}%
725   \ifundef\c@wrglossary
726   {%
727     \newcounter{wrglossary}%
728     \renewcommand{\thewrglossary}{\arabic{wrglossary}}%
729   }%
730   {}%
731   \renewcommand*{\glxstr@inc@wrglossaryctr}[1]{%
```

Only increment if the current counter is wrglossary.

```
732   \ifdefstring\@glsc@counter{wrglossary}%
733   {%
734     \refstepcounter{wrglossary}%
735     \label{wrglossary.\thewrglossary}%
736   }%
737   {}%
738 }%
739 \renewcommand*{\GlsXtrInternalLocationHyperlink}[3]{%
740   \ifdefstring\glsc@entrycounter{wrglossary}%
741   {%
742     \@glxstr@wrglossary@locationhyperlink{##1}{##2}{##3}%
743   }%
744   {\glxstr@hyperlink{##1##2##3}{##3}}%
745   }%
746 }
```

sxtrwrglossmark

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

```
747 \newcommand*{\@glxstrwrglossmark}{}
```

sxtrwrglossmark

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

```
748 \newcommand*{\@glxstrwrglossmark}{}
749 \AtBeginDocument{\renewcommand*{\@glxstrwrglossmark}{\@glxstrwrglossmark}}
```

sxtrwrglossmark

Does nothing by default.

```
750 \newcommand*{\glxstrwrglossmark}{\ensuremath{\cdot}}
```

debug Provide extra debug options.

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

Pass all other options to glossaries.

```
772 \DeclareOptionX*{%
773   \expandafter\glstr@doption\expandafter{\CurrentOption}}
```

Process options.

```
774 \ProcessOptionsX
```

Load glossaries if not already loaded.

```
775 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
776 \@glstr@doaccsupp
```

Load the glossaries-prefix package if required.

```
777 \@glstr@doloadprefix
```

Redefine \glspostdescription if required.

```
778 \@glstr@defpostpunc
```

\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.

```
779 \ifdef\glsshowtargetouter
780 {
781   \renewcommand*{\glsshowtarget}[1]{%
782     \glstr@titleorpdforheading
783     {%
784       \ifmmode
```

```

785     \nfss@text{\glsshowtargetfont [#1]}%
786   \else
787     \ifinner
788       {\glsshowtargetfont [#1]}%
789     \else
790       \glsshowtargetouter{#1}%
791     \fi
792   \fi
793 }%
794 {[#1]}%
795 {\protect\glsshowtargetfont [#1]}%
796 }
797 }
798 {

```

Old definition.

```

799 \def\glsshowtarget#1{%
800   \glstrtitleorpdforheading
801   {%
802     \ifmmode
803       \texttt{\small [#1]}%
804     \else
805       \ifinner
806         \texttt{\small [#1]}%
807       \else
808         \marginpar{\texttt{\small #1}}%
809       \fi
810     \fi
811   }%
812   {[#1]}%
813   {\texttt{\small [#1]}%
814 }
815 }

```

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

```

816 \let\@glxtr@org@doseeglossary\@do@seeglossary

```

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

```

817 \newcommand*{\@glxtr@doseeglossary}[2]{%
818   \glsdofexists{#1}%
819   {%
820     \@glxtrwrglossmark
821     \@glxtr@org@doseeglossary{#1}{#2}%
822   }%
823 }

```

oindex@glossary

```

824 \newcommand*{\@glxtr@dosee@alsoindex@glossary}[2]{%
825   \@glxtr@recordsee{#1}{#2}%

```

```

826 \@glxtr@doseeglossary{#1}{#2}%
827 }

@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.)
828 \let\@glxtr@org@gloautosee\@glo@autosee

Check if user tried autoseeindex=false when it can't be supported.
829 \if@glxtr@autoseeindex
830 \else
831 \ifdef\@glxtr@org@gloautosee
832 {}%
833 {\PackageError{glossaries-extra}{‘autoseeindex=false’ package
834 option requires at least v4.30 of glossaries.sty}%
835 {You need to update the glossaries.sty package}%
836 }
837 \fi

\@glo@autosee If \@glo@autosee has been defined (glossaries v4.30 onwards), redefine it to test the au-
toseeindex option.
838 \ifdef\@glo@autosee
839 {%
840 \renewcommand*{\@glo@autosee}{%
841 \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
842 }%
843 {}

checkseeallowed Don't prohibit the use of the see key before the indexing files have been opened if the auto-
matic see indexing has been disabled, since it's no longer an issue.
844 \renewcommand*{\gls@checkseeallowed}{%
845 \if@glxtr@autoseeindex\@gls@see@noindex\fi
846 }

Define abbreviations glossaries if required.
847 \@glxtr@abbreviationsdef
848 \let\@glxtr@abbreviationsdef\relax

Setup shortcuts if required.
849 \@glxtr@setupshortcuts

Redefine \@glxtr@redef@for@gl@sentries if required.
850 \@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:
851 \renewcommand{\@glxtr@doooption}[1]{\@setupglossaries{#1}}%

```

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

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

Now define the user command:

```
853 \newcommand*{\glossariesextrasetup}[1]{%
854   \let\glsxtr@setup@record\relax
855   \let\glsxtr@setupshortcuts\relax
856   \let\glsxtr@redef@forglsentries\relax
857   \let\glsxtr@doloadprefix\relax
858   \setkeys{glossaries-extra.sty}{#1}%
859   \@glsxtr@abbreviationsdef
860   \let\glsxtr@abbreviationsdef\relax
861   \@glsxtr@setupshortcuts
862   \glsxtr@setup@record
863   \glsxtr@redef@forglsentries
864   \@glsxtr@doloadprefix
865 }
```

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

```
866 \let\glsxtr@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.

```
867 \newcommand*{\glsxtr@@do@wrglossary}[1]{%
868   \@glsxtrwrglossmark
869   \glsxtr@inc@wrglossaryctr{#1}%
870   \glsxtr@org@@do@wrglossary{#1}%
871 }
```

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

```
872 \let\glsxtr@saveentrycounter\@gls@saveentrycounter
```

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

```
873 \let\@gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
```

`getcounterprefix` 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.

```
874 \renewcommand*\@gls@getcounterprefix[2]{%
875   \protected@edef\@gls@thisloc{#1}\protected@edef\@gls@thisHloc{#2}%
876   \ifx\@gls@thisloc\@gls@thisHloc
877     \def\@glo@counterprefix{}%
878   \else
879     \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
880       \def\@glo@tmp{##2}%
```

```

881 \ifx\@glo@tmp\@empty
882 \def\@glo@counterprefix{}%
883 \else
884 \def\@glo@counterprefix{##1}%
885 \fi
886 }%
887 \@gls@get@counterprefix#2.#1\end@getprefix
Warn if no prefix can be formed, unless record=nameref.
888 \ifx\@glo@counterprefix\@empty
889 \ifx\@glsxtr@record@setting\@glsxtr@record@setting@nameref
890 \else
891 \GlossariesExtraWarning{Hyper target ‘#2’ can’t be formed by
892 prefixing^^Jlocation ‘#1’. You need to modify the
893 definition of \string\theH\@gls@counter^^Jotherwise you
894 will get the warning: “name{\@gls@counter.#1}’ has been^^J
895 referenced but does not exist”%
896 \ifx\@glsxtr@record@setting\@glsxtr@record@setting@only
897 . You may want to consider using record=nameref instead%
898 \fi}%
899 \fi
900 \fi
901 \fi
902 }

```

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

sxtrdialecthook

```
903 \newcommand*{\@glsxtrdialecthook}{}

```

Set up record option if required.

```
904 \glsxtr@setup@record

```

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

```

905 \AtBeginDocument{%
906 \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
907 \def\@glsxtrundeftag{\glsxtrundeftag}%
908 }

```

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).


```

909 \newcommand*{\GlsXtrIfUnusedOrUndefined}[3]{%
910   \ifglentryexists{#1}%
911   {\ifbool{glo@\glsdetoklabel{#1}@flag}{#3}{#2}}%
912   {#2}}%
913 }

```

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

```

914 \ifdef\s@ifglossaryexists
915 {}
916 {

```

`\ifglossaryexists`

```

917   \renewcommand{\ifglossaryexists}{%
918     \@ifstar\s@ifglossaryexists\@ifglossaryexists
919   }

```

`\ifglossaryexists`

```

920   \newcommand{\@ifglossaryexists}[3]{%
921     \ifcsundef{glo@#1@out}{#3}{#2}%
922   }

```

`\ifglossaryexists`

```

923   \newcommand{\s@ifglossaryexists}[3]{%
924     \ifcsundef{glolist@#1}{#3}{#2}%
925   }
926 }

```

`\ifemptyglossary`

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

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

```

927 \newcommand{\glstrifemptyglossary}[3]{%
928   \ifcsdef{glolist@#1}%
929   {%
930     \ifcsstring{glolist@#1}{,}{#2}{#3}%
931   }%
932   {%
933     \glstrundefaction{Glossary type '#1' doesn't exist}{}%
934     #2%
935   }%
936 }

```

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

```
937 \newcommand*{\glxtrifkeydefined}[3]{%
938   \key@ifundefined{glossentry}{#1}{#3}{#2}%
939 }
```

providestoragekey Like \gl saddstoragekey but does nothing if the key has already been defined.

```
940 \newcommand*{\glxtrprovidestoragekey}{%
941   \@ifstar\@sglsxtr@provide@storagekey\@glxtr@provide@storagekey
942 }
```

vide@storagekey Unstarred version.

```
943 \newcommand*{\@glxtr@provide@storagekey}[3]{%
944   \key@ifundefined{glossentry}{#1}%
945   {%
946     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
947     \appto\@gls@keymap{,{#1}{#1}}%
948     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
949     \appto\@newglossaryentryposthook{%
950       \letcs{\@glo@tmp}{@glo@#1}%
951       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
952     }%

```

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

```
953   \ifblank{#3}
954   {}%
955   {%
956     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
957   }%
958 }%
959 {%

```

Provide the no-link command if not already defined.

```
960   \ifblank{#3}
961   {}%
962   {%
963     \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
964   }%
965 }%
966 }
```

vide@storagekey Starred version.

```
967 \newcommand*{\s@glxtr@provide@storagekey}[1]{%
968   \key@ifundefined{glossentry}{#1}%
969   {%
970     \expandafter\newcommand\expandafter*\expandafter
971     {\csname gls@assign@#1@field\endcsname}[2]{%
972       \@gls@expand@field{##1}{#1}{##2}%
973     }%

```

```

974 }%
975 {}%
976 \@glxstr@provide@addstoragekey{#1}%
977 }

```

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

```

978 \newcommand{\GlsXtrFmtField}{useri}

```

`\glxstrfmt`

```

979 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

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

```

980 \newrobustcmd*{\glxstrfmt}{\@ifstar\s@glxstrfmt\@glxstrfmt}

```

`\@glxstrfmt` Unstarred form.

```

981 \newcommand*{\@glxstrfmt}[3][\@glxstrfmt{#1}{#2}{#3}]{}

```

`\s@glxstrfmt` Starred form.

```

982 \newcommand*{\s@glxstrfmt}[3][\@glxstrfmt{#1}{#2}{#3}]{}
983 \new@ifnextchar[\s@glxstrfmt{#1}{#2}{#3}]{}
984 {\@glxstrfmt{#1}{#2}{#3}]{}
985 }

```

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

```

986 \def\s@glxstrfmt#1#2#3[#4]{\@glxstrfmt{#1}{#2}{#3}{#4}}

```

`\@glxstrfmt` Actual inner working.

```

987 \newcommand*{\@glxstrfmt}[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).

```

988 \begingroup
989 \def\glslabel{#2}%
990 \glsdoifexistsordo{#2}%
991 {%
992   \ifglshasfield{\GlsXtrFmtField}{#2}%
993   {%
994     \let\do@gl@link@checkfirsthyper\relax
995     \expandafter\@gl@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
996     {\glxstrfmtdisplay{\glscurrentfieldvalue}{#3}{#4}}%
997   }%
998   {\glxstrfmtdisplay{@firstofone}{#3}{#4}}%
999 }%
1000 {%

```

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

```

1001 \begingroup
1002 \gls@setdefault@glslink@opts
1003 \setkeys{glslink}{\GlsXtrFmtDefaultOptions,#1}%
1004 \ifKV@glslink@noindex\else\glsadd{#2}\fi
1005 \endgroup
1006 \glsxtrfmtdisplay{@firstofone}{#3}{#4}%
1007 }%
1008 \endgroup
1009 }

```

`\glsxtrfmtdisplay` The command used internally by `\glsxtrfmt` 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).

```

1010 \newcommand{\glsxtrfmtdisplay}[3]{\csuse{#1}{#2}#3}

```

`\glstxrentryfmt` No link or indexing.

```

1011 \ifdef\texorpdfstring
1012 {
1013 \newcommand*{\glstxrentryfmt}[2]{%
1014 \texorpdfstring{@\glstxrentryfmt{#1}{#2}}{\glstxtrpdfentryfmt{#1}{#2}}%
1015 }
1016 }
1017 {
1018 \newcommand*{\glstxrentryfmt}{\@glstxrentryfmt}
1019 }

```

`\glstxtrpdfentryfmt` Use for the PDF bookmarks.

```

1020 \newcommand*{\glstxtrpdfentryfmt}[2]{#2}

```

`@glstxrentryfmt`

```

1021 \newrobustcmd*{@\glstxrentryfmt}[2]{%

```

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

```

1022 {%
1023 \protected@edef\glslabel{#1}%
1024 \glsdoifexistsordo{#1}%
1025 {%
1026 \ifglshasfield{\GlsXtrFmtField}{#1}%
1027 {%
1028 \csuse{\glscurrentfieldvalue}{#2}%
1029 }%
1030 {#2}%
1031 }%
1032 {#2}%
1033 }%
1034 }

```

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

```
1035 \newcommand*\glstrfieldlistadd}[3]{%
1036   \listcsadd{glo@\glstetoklabel{#1}@\#2}{#3}%
1037 }
```

`trfieldlistgadd` Similarly but uses `\listcsgadd`.

```
1038 \newcommand*\glstrfieldlistgadd}[3]{%
1039   \listcsgadd{glo@\glstetoklabel{#1}@\#2}{#3}%
1040 }
```

`trfieldlisteadd` Similarly but uses `\listcseadd`.

```
1041 \newcommand*\glstrfieldlisteadd}[3]{%
1042   \listcseadd{glo@\glstetoklabel{#1}@\#2}{#3}%
1043 }
```

`trfieldlistxadd` Similarly but uses `\listcsxadd`.

```
1044 \newcommand*\glstrfieldlistxadd}[3]{%
1045   \listcsxadd{glo@\glstetoklabel{#1}@\#2}{#3}%
1046 }
```

Now provide commands to iterate over these lists.

`fielddolistloop`

```
1047 \newcommand*\glstrfielddolistloop}[2]{%
1048   \dolistcsloop{glo@\glstetoklabel{#1}@\#2}%
1049 }
```

`fieldforlistloop`

```
1050 \newcommand*\glstrfieldforlistloop}[3]{%
1051   \forlistcsloop{#3}{glo@\glstetoklabel{#1}@\#2}%
1052 }
```

`fieldformatlist`

```
1053 \newrobustcmd*\glstrfieldformatlist}[2]{%
1054   \begingroup
1055   \def\@dtl@formatlist@itemsep{}%
1056   \def\@dtl@formatlist@lastitem{}%
1057   \def\@dtl@formatlist@prelastitem{}%
1058   \def\@dtl@formatlist@prelastitemsep{}%
1059   \forlistcsloop{\@dtl@formatlist@handler}{glo@\glstetoklabel{#1}@\#2}%
1060   \@dtl@formatlist@prelastitem\@dtl@formatlist@lastitem
1061   \endgroup
1062 }
```

List element tests:

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

```
1063 \newcommand*{\glstrfieldifinlist}[5]{%
1064   \ifinlistcs{#3}{glo@\glsetoklabel{#1}@#2}{#4}{#5}%
1065 }
```

rfieldxifinlist Expands item.

```
1066 \newcommand*{\glstrfieldxifinlist}[5]{%
1067   \xifinlistcs{#3}{glo@\glsetoklabel{#1}@#2}{#4}{#5}%
1068 }
```

sxtrforcsvfield

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

```
1069 \newcommand*{\glsxtrforcsvfield}[3]{%
1070   \@glxtrifhasfield{#2}{#1}%
1071   {%
1072     \let\glxtrendfor\@endfortrue
1073     \@for\@glxtr@label:=\glscurrentfieldvalue\do
1074       {\expandafter#3\expandafter{\@glxtr@label}}}%
1075   }%
1076 }
```

ldformatcsvlist

```
1077 \newrobustcmd*{\glstrfieldformatcsvlist}[2]{%
1078   \@glxtrifhasfield{#2}{#1}%
1079   {\@dtlformatlist\glscurrentfieldvalue}%
1080   }%
1081 }
```

dValueInCsvList

`\GlsXtrIfFieldValueInCsvList{<label>}{<field>}{<list>}{<true>}{<false>}`

```
1082 \newcommand*{\GlsXtrIfFieldValueInCsvList}{%
1083   \@ifstar\s@GlsXtrIfFieldValueInCsvList\@GlsXtrIfFieldValueInCsvList
1084 }
```

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

dValueInCsvList Unstarred version.

```
1085 \newcommand*{\@GlsXtrIfFieldValueInCsvList}[5]{%
1086   \@glxtrifhasfield{#2}{#1}%
1087   {%
1088     \expandafter\DTLifinlist\expandafter{\glscurrentfieldvalue}%

```

```

1089   {#3}{#4}{#5}%
1090 }%
1091 {#5}%
1092 }

```

dValueInCsvList Starred version.

```

1093 \newcommand*{\s@GlsXtrIfFieldValueInCsvList}[5]{%
1094   \s@glstrifhasfield{#2}{#1}%
1095   {%
1096     \expandafter\DTLifinlist\expandafter{\glscurrentfieldvalue}%
1097     {#3}{#4}{#5}%
1098   }%
1099   {#5}%
1100 }

```

lsxtrifhasfield 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.

```

1101 \newrobustcmd{\glstrifhasfield}{%
1102   \@ifstar{\s@glstrifhasfield}{\@glstrifhasfield}%
1103 }

```

lsxtrifhasfield Unstarred version adds grouping.

```

1104 \newcommand{\@glstrifhasfield}[4]{%
1105   {\s@glstrifhasfield{#1}{#2}{#3}{#4}}%
1106 }

```

lsxtrifhasfield Starred version omits grouping.

```

1107 \newcommand{\s@glstrifhasfield}[4]{%
1108   \letcs{\glscurrentfieldvalue}{glo@\glsdetoklabel{#2}@#1}%
1109   \ifundef\glscurrentfieldvalue
1110     {#4}%
1111   {%
1112     \ifdefempty\glscurrentfieldvalue{#4}{#3}%
1113   }%
1114 }

```

rIfFieldNonZero Designed for numeric fields.

```

1115 \newcommand{\GlsXtrIfFieldNonZero}{%
1116   \@ifstar\s@GlsXtrIfFieldNonZero\@GlsXtrIfFieldNonZero
1117 }

```

rIfFieldNonZero

```

1118 \newcommand{\@GlsXtrIfFieldNonZero}[4]{%
1119   \@GlsXtrIfFieldCmpNum{#1}{#2}{=}{0}{#4}{#3}%
1120 }

```

XtrIfFieldEqNum

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

Designed for numeric fields.

```
1121 \newcommand{\GlsXtrIfFieldEqNum}{%
1122   \ifstar\s@GlsXtrIfFieldEqNum\@GlsXtrIfFieldEqNum
1123 }
```

XtrIfFieldEqNum

```
1124 \newcommand{\@GlsXtrIfFieldEqNum}[5]{%
1125   \@GlsXtrIfFieldCmpNum{#1}{#2}{=}{#3}{#4}{#5}%
1126 }
```

XtrIfFieldEqNum

```
1127 \newcommand{\s@GlsXtrIfFieldEqNum}[5]{%
1128   \s@GlsXtrIfFieldCmpNum{#1}{#2}{=}{#3}{#4}{#5}%
1129 }
```

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.

```
1130 \newcommand{\GlsXtrIfFieldCmpNum}{%
1131   \ifstar\s@GlsXtrIfFieldCmpNum\@GlsXtrIfFieldCmpNum
1132 }
```

trIfFieldCmpNum

```
1133 \newcommand{\@GlsXtrIfFieldCmpNum}[6]{%
1134   {%
1135     \letcs\glscurrentfieldvalue{glo@\glsetoklabel{#2}@#1}%
1136     \ifundef\glscurrentfieldvalue
1137       {\def\glscurrentfieldvalue{0}}%
1138     {%
1139       \ifdefempty\glscurrentfieldvalue
1140         {\def\glscurrentfieldvalue{0}}%
1141       }%
1142     }%
1143     \ifnum\glscurrentfieldvalue#3#4\relax #5\else #6\fi
1144   }%
1145 }
```

trIfFieldCmpNum

```
1146 \newcommand{\s@GlsXtrIfFieldCmpNum}[6]{%
```



```

1147 \letcs{\glscurrentfieldvalue}{glo@\glsdetoklabel{#2}@#1}%
1148 \ifundef\glscurrentfieldvalue
1149 {\def\glscurrentfieldvalue{0}}%
1150 {%
1151 \ifdefempty\glscurrentfieldvalue
1152 {\def\glscurrentfieldvalue{0}}%
1153 {}%
1154 }%
1155 \ifnum\glscurrentfieldvalue#3#4\relax #5\else #6\fi
1156 }

```

XtrIfFieldUndef

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

Just uses `\ifcsundef`.

```

1157 \newcommand{\GlsXtrIfFieldUndef}[2]{%
1158 \ifcsundef{glo@\glsdetoklabel{#2}@#1}%
1159 }

```

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

```

1160 \newcommand*{\glsxtrusefield}[2]{%
1161 \@gls@entry@field{#1}{#2}%
1162 }

```

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

```

1163 \ifdef\texorpdfstring
1164 {
1165 \newcommand*{\Glsxtrusefield}[2]{%
1166 \texorpdfstring
1167 {\@Gls@entry@field{#1}{#2}}
1168 {\@gls@entry@field{#1}{#2}}%
1169 }
1170 }
1171 {
1172 \newcommand*{\Glsxtrusefield}[2]{%
1173 \@Gls@entry@field{#1}{#2}%
1174 }
1175 }

```

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

```

1176 \ifdef\texorpdfstring
1177 {
1178 \newcommand*{\GLSxtrusefield}[2]{%
1179 \texorpdfstring
1180 {\glsdoifexists{#1}{\mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}}}%

```

```

1181      {\@gls@entry@field{#1}{#2}}%
1182    }
1183 }
1184 {
1185   \newcommand*{\GLSxtrusefield}[2]{%
1186     \glsdoifexists{#1}{\mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}}%
1187   }
1188 }

```

entryparentname

```

1189 \newcommand*{\glstrentryparentname}[1]{%
1190   \ifcsdef{glo@\glsdetoklabel{#1}@parent}%
1191     {\csuse{glo@\csuse{glo@\glsdetoklabel{#1}@parent}@name}}%
1192   }%
1193 }

```

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

```

1194 \newcommand*{\glsxtrdeffield}[2]{\csdef{glo@\glsdetoklabel{#1}@#2}}

```

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

```

1195 \newcommand*{\glsxtredeffield}[2]{\protected@csedef{glo@\glsdetoklabel{#1}@#2}}

```

etfieldifexists

```

1196 \newcommand*{\glsxtrsetfieldifexists}[3]{\glsdoifexists{#1}{#3}}

```

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

```

1197 \newrobustcmd*{\GlsXtrSetField}[3]{%
1198   \glsxtrsetfieldifexists{#1}{#2}%
1199   {\csdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
1200 }

```

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

```

1201 \newrobustcmd*{\GlsXtrLetField}[3]{%
1202   \glsxtrsetfieldifexists{#1}{#2}%
1203   {\cslet{glo@\glsdetoklabel{#1}@#2}{#3}}%
1204 }

```

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

```

1205 \newrobustcmd*{\csGlsXtrLetField}[3]{%
1206   \glsxtrsetfieldifexists{#1}{#2}%
1207   {\csletcs{glo@\glsdetoklabel{#1}@#2}{#3}}%
1208 }

```

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.

```

1209 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
1210   \glsxtrsetfieldifexists{#1}{#2}%

```

```

1211 {\csletcs{glo@\glsdetoklabel{#1}@\#2}{glo@\glsdetoklabel{#3}@\#4}}%
1212 }

```

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

```

1213 \newrobustcmd*{\gGlsXtrSetField}[3]{%
1214   \glsxtrsetfieldifexists{#1}{#2}%
1215   {\csgdef{glo@\glsdetoklabel{#1}@\#2}{#3}}%
1216 }

```

xGlsXtrSetField

```

1217 \newrobustcmd*{\xGlsXtrSetField}[3]{%
1218   \glsxtrsetfieldifexists{#1}{#2}%
1219   {\protected@csxdef{glo@\glsdetoklabel{#1}@\#2}{#3}}%
1220 }

```

eGlsXtrSetField

```

1221 \newrobustcmd*{\eGlsXtrSetField}[3]{%
1222   \glsxtrsetfieldifexists{#1}{#2}%
1223   {\protected@csedef{glo@\glsdetoklabel{#1}@\#2}{#3}}%
1224 }

```

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

```

1225 \newcommand*{\GlsXtrIfFieldEqStr}{%
1226   \@ifstar\s@GlsXtrIfFieldEqStr\@GlsXtrIfFieldEqStr
1227 }

```

XtrIfFieldEqStr

```

1228 \newrobustcmd*{\@GlsXtrIfFieldEqStr}[5]{%
1229   \@glsxtrifhasfield{#1}{#2}%
1230   {%
1231     \ifdefstring{\glscurrentfieldvalue}{#3}{#4}{#5}%
1232   }%
1233   {#5}%
1234 }

```

XtrIfFieldEqStr

```

1235 \newrobustcmd*{\s@GlsXtrIfFieldEqStr}[5]{%
1236   \s@glsxtrifhasfield{#1}{#2}%
1237   {%
1238     \ifdefstring{\glscurrentfieldvalue}{#3}{#4}{#5}%
1239   }%
1240   {#5}%
1241 }

```

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

```

1242 \newcommand*{\GlsXtrIfFieldEqXpStr}{%

```

```

1243 \@ifstar\s@GlsXtrIfFieldEqXpStr\@GlsXtrIfFieldEqXpStr
1244 }

```

rIfFieldEqXpStr

```

1245 \newrobustcmd*{\@GlsXtrIfFieldEqXpStr}[5]{%
1246 \@glstrifhasfield{#1}{#2}%
1247 {%
1248 \protected@edef\@gls@tmp{#3}%
1249 \ifdefequal{glscurrentfieldvalue}{\@gls@tmp}{#4}{#5}%
1250 }%
1251 {#5}%
1252 }

```

rIfFieldEqXpStr

```

1253 \newrobustcmd*{\s@GlsXtrIfFieldEqXpStr}[5]{%
1254 \s@glstrifhasfield{#1}{#2}%
1255 {%
1256 \protected@edef\@gls@tmp{#3}%
1257 \ifdefequal{glscurrentfieldvalue}{\@gls@tmp}{#4}{#5}%
1258 }%
1259 {#5}%
1260 }

```

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

```

1261 \newcommand*{\GlsXtrIfXpFieldEqXpStr}{%
1262 \@ifstar\s@GlsXtrIfXpFieldEqXpStr\@GlsXtrIfXpFieldEqXpStr
1263 }

```

fXpFieldEqXpStr

```

1264 \newrobustcmd*{\@GlsXtrIfXpFieldEqXpStr}[5]{%
1265 \@glstrifhasfield{#1}{#2}%
1266 {%
1267 \protected@edef\@gls@tmp{glscurrentfieldvalue}%
1268 \let\glscurrentfieldvalue\@gls@tmp
1269 \protected@edef\@gls@tmp{#3}%
1270 \ifdefequal{glscurrentfieldvalue}{\@gls@tmp}{#4}{#5}%
1271 }%
1272 {#5}%
1273 }

```

fXpFieldEqXpStr

```

1274 \newrobustcmd*{\s@GlsXtrIfXpFieldEqXpStr}[5]{%
1275 \s@glstrifhasfield{#1}{#2}%
1276 {%
1277 \protected@edef\@gls@tmp{glscurrentfieldvalue}%
1278 \let\glscurrentfieldvalue\@gls@tmp
1279 \protected@edef\@gls@tmp{#3}%

```

```

1280 \ifdefequal{\glscurrentfieldvalue}{\@gls@tmp}{#4}{#5}%
1281 }%
1282 {#5}%
1283 }

```

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`.

```

1284 \ifdef\foreignlanguage
1285 {
1286 \ifdef\GetTrackedDialectFromLanguageTag
1287 {
1288 \newcommand{\GlsXtrForeignText}[2]{%

```

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

```

1289 \let\@glxtr@org@currentfieldvalue\glscurrentfieldvalue
1290 \glsxtrifhasfield{\GlsXtrForeignTextField}{#1}%
1291 {%
1292 \expandafter\GetTrackedDialectFromLanguageTag\expandafter
1293 {\glscurrentfieldvalue}{\@glxtr@dialect}%
1294 \let\@glxtr@locale\glscurrentfieldvalue
1295 \let\glscurrentfieldvalue\@glxtr@org@currentfieldvalue
1296 \ifdefempty\@glxtr@dialect
1297 {%

```

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

```

1298 \ifundef\TrackedDialectClosestSubMatch
1299 {%
1300 \GlossariesExtraWarning{Can't obtain dialect label
1301 (tracklang v1.3.6+ required)}%
1302 }%
1303 {\let\@glxtr@dialect\TrackedDialectClosestSubMatch}%
1304 }%
1305 {}%
1306 \ifdefempty\@glxtr@dialect
1307 {%

```

No tracked dialect found for the root language.

```

1308 }%
1309 {%

```

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

```

1310 \ifcsundef{captions\@glxtr@dialect}{}%
1311 {%

```

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

```

1312      \IfTrackedDialectHasMapping{\@glsxtr@dialect}%
1313      {%
1314      \edef\@glsxtr@dialect{%
1315      \GetTrackedDialectToMapping{\@glsxtr@dialect}}%

```

Does a caption hook exist for this?

```

1316      \ifcsundef{captions\@glsxtr@dialect}{}%
1317      {%

```

No mapping. Try root language label instead.

```

1318      \ifcsundef{captions\@tracklang@lang}{}%
1319      {%
1320      \let\@glsxtr@dialect\@tracklang@lang
1321      }%
1322      }%
1323      }%
1324      {%

```

No mapping. Try root language label instead.

```

1325      \ifcsundef{captions\@tracklang@lang}{}%
1326      {%
1327      \let\@glsxtr@dialect\@tracklang@lang
1328      }%
1329      }%
1330      }%
1331      }%
1332      \ifdefempty\@glsxtr@dialect
1333      {%
1334      \GlsXtrUnknownDialectWarning{\@glsxtr@locale}{\@tracklang@lang}%
1335      #2%
1336      }%
1337      {\foreignlanguage{\@glsxtr@dialect}{#2}}%
1338      }%
1339      {#2}% key not set
1340      }
1341      }
1342      {
1343      \newcommand{\GlsXtrForeignText}[2]{%
1344      \GlossariesExtraWarning{Can't encapsulate foreign text:
1345      tracklang v1.3.6+ required}%
1346      #2%
1347      }
1348      }
1349      }
1350      {

```

\foreignlanguage isn't defined so just do *<text>*.

```

1351      \newcommand{\GlsXtrForeignText}[2]{#2}
1352      }

```

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

```
1353 \newcommand*{\GlsXtrForeignTextField}{userii}
```

nDialectWarning

```
1354 \newcommand*{\GlsXtrUnknownDialectWarning}[2]{%
1355   \GlossariesExtraWarning{Can't determine valid dialect label
1356   for locale '#1' (root language: #2)}%
1357 }
```

\glstrpageref Like \glsrefentry 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.

```
1358 \ifdef\GlsEntryCounterLabelPrefix
1359 {%
1360   \newcommand*{\glstrpageref}[1]{%
1361     \ifglssentrycounter
1362       \pageref{\GlsEntryCounterLabelPrefix\glsdetoklabel{#1}}%
1363     \else
1364       \ifglssubentrycounter
1365         \pageref{\GlsEntryCounterLabelPrefix\glsdetoklabel{#1}}%
1366       \else
1367         \gls{#1}%
1368       \fi
1369     \fi
1370   }
1371 }%
1372 {%
1373   \newcommand*{\glstrpageref}[1]{%
1374     \ifglssentrycounter
1375       \pageref{glsentry-\glsdetoklabel{#1}}%
1376     \else
1377       \ifglssubentrycounter
1378         \pageref{glsentry-\glsdetoklabel{#1}}%
1379       \else
1380         \gls{#1}%
1381       \fi
1382     \fi
1383   }
1384 }%
```

lossarypreamble

```
1385 \newcommand{\apptoglossarypreamble}[2][\glsdefaulttype]{%
1386   \ifcsdef{glolist@#1}%
1387   {%
1388     \ifcsundef{@glossarypreamble@#1}%
1389     {\csdef{@glossarypreamble@#1}{}}%
1390   }%
1391   \csappto{@glossarypreamble@#1}{#2}%

```

```

1392 }%
1393 {%
1394   \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
1395 }%
1396 }

```

lossarypreamble

```

1397 \newcommand{\preglossarypreamble}[2][\glsdefaulttype]{%
1398   \ifcsdef{glolist@#1}%
1399   {%
1400     \ifcsundef{@glossarypreamble@#1}%
1401     {\csdef{@glossarypreamble@#1}{}}%
1402     {}%
1403     \cspretoto{@glossarypreamble@#1}{#2}%
1404   }%
1405   {%
1406     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
1407   }%
1408 }

```

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.

```

1409 \ifdef\@gls@entry@field
1410 {
1411   \renewcommand*{\@gls@entry@field}[2]{\csuse{glo@glstoklabel{#1}@#2}}
1412 }
1413 {}

```

`\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`.

```
1414 \renewcommand*{\ifglsused}[3]{%
1415   \glsdoifexists{#1}{\ifbool{glo@\glsdetoklabel{#1}@flag}{#2}{#3}}%
1416 }
```

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

`\newglossaryentry`

```
1417 \renewcommand*{\longnewglossaryentry}{%
1418   \@ifstar{\glsxtr@s@longnewglossaryentry\glsxtr@longnewglossaryentry
1419 }
```

`\newglossaryentry` Starred version.

```
1420 \newcommand{\@glsxtr@s@longnewglossaryentry}[3]{%
1421   \glsdoifnoexists{#1}%
1422   {%
1423     \bgroup
1424     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1425     \long\def\@newglossaryentryprehook{%
1426       \long\def\@glo@desc{#3}%
1427       \@org@newglossaryentryprehook
1428     }%
1429     \renewcommand*{\gls@assign@desc}[1]{%
1430       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
1431       \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
1432     }
1433     \gls@defglossaryentry{#1}{#2}%
1434   \egroup
1435   }%
1436 }
```

`\newglossaryentry` Unstarred version.

```
1437 \newcommand{\@glsxtr@longnewglossaryentry}[3]{%
1438   \glsdoifnoexists{#1}%
1439   {%
1440     \bgroup
1441     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1442     \long\def\@newglossaryentryprehook{%
1443       \long\def\@glo@desc{#3\glsxtrpostlongdescription}%
1444       \@org@newglossaryentryprehook
1445     }%
1446     \renewcommand*{\gls@assign@desc}[1]{%
1447       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
1448     }
```

The following is different from the base glossaries.sty:

```

1448      \global\cslet{glo@glstoklabel{#1}@descplural}{\@glo@descplural}%
1449      }
1450      \gls@defglossaryentry{#1}{#2}%
1451      \egroup
1452    }%
1453  }

```

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

```

1454 \newcommand*{\glstrpostlongdescription}{\leavevmode\unskip\nopostdesc}

```

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

`\ignoredglossary` Redefine to check for star.

```

1455 \renewcommand{\newignoredglossary}{%
1456   \@ifstar\glstr@s@newignoredglossary\glstr@org@newignoredglossary
1457 }

```

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

```

1458 \newcommand*{\glstr@org@newignoredglossary}[1]{%
1459   \ifcsdef{glolist@#1}
1460   {%
1461     \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1462   }%
1463   {%
1464     \ifdefempty\@ignored@glossaries
1465     {%
1466       \protected@edef\@ignored@glossaries{#1}%
1467     }%
1468     {%
1469       \protected@eappto\@ignored@glossaries{,#1}%
1470     }%
1471     \csgdef{glolist@#1}{,}%
1472     \ifcsundef{gls@#1@entryfmt}%
1473     {%
1474       \defglentryfmt[#1]{\glentryfmt}%
1475     }%
1476     {}%
1477     \ifdefempty\@gls@nohyperlist
1478     {%
1479       \renewcommand*{\@gls@nohyperlist}{#1}%
1480     }%
1481     {%
1482       \protected@eappto\@gls@nohyperlist{,#1}%
1483     }%
1484   }%
1485 }

```

ignoredglossary Starred form.

```
1486 \newcommand*{\glxtr@s@newignoredglossary}[1]{%
1487   \ifcsdef{glolist@#1}
1488   {%
1489     \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
1490   }%
1491   {%

1492   \ifdefempty\@ignored@glossaries
1493   {%
1494     \protected@edef\@ignored@glossaries{#1}%
1495   }%
1496   {%
1497     \protected@eappto\@ignored@glossaries{,#1}%
1498   }%
1499   \csgdef{glolist@#1}{,}%
1500   \ifcsundef{gls@#1@entryfmt}%
1501   {%
1502     \defglsentryfmt[#1]{\glsentryfmt}%
1503   }%
1504   {}%
1505 }%
1506 }
```

\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.

```
1507 \glsifusetranslator
1508 {%
1509   \renewcommand*{\glsettoctitle}[1]{%
1510     \ifcsdef{gls@tr@set@#1@toctitle}%
1511     {%
1512       \csuse{gls@tr@set@#1@toctitle}%
1513     }%
1514     {%
1515       \ifcsdef{@glotype@#1@title}%
1516       {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1517       {\def\glossarytoctitle{\glossarytitle}}%
1518     }%
1519   }%
1520 }
1521 {
1522   \renewcommand*{\glsettoctitle}[1]{%
1523     \ifcsdef{@glotype@#1@title}%
1524     {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1525     {\def\glossarytoctitle{\glossarytitle}}%
1526   }
1527 }
```

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

```

1528 \newcommand{\provideignoredglossary}{%
1529 \ifstar\glxtr@s@provideignoredglossary\glxtr@provideignoredglossary
1530 }

```

ignoredglossary Unstarred version.

```

1531 \newcommand*{\glxtr@provideignoredglossary}[1]{%
1532 \ifcsdef{glolist@#1}
1533 {}%
1534 {%
1535 \ifdefempty\@ignored@glossaries
1536 {%
1537 \protected@edef\@ignored@glossaries{#1}%
1538 }%
1539 {%
1540 \protected@eappto\@ignored@glossaries{,#1}%
1541 }%
1542 \csgdef{glolist@#1}{,}%
1543 \ifcsundef{gls@#1@entryfmt}%
1544 {%
1545 \defgl@sentryfmt[#1]{\gl@sentryfmt}%
1546 }%
1547 {}%
1548 \ifdefempty\@gls@nohyperlist
1549 {%
1550 \renewcommand*{\@gls@nohyperlist}{#1}%
1551 }%
1552 {%
1553 \protected@eappto\@gls@nohyperlist{,#1}%
1554 }%
1555 }%
1556 }

```

ignoredglossary Starred form.

```

1557 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
1558 \ifcsdef{glolist@#1}
1559 {}%
1560 {%
1561 \ifdefempty\@ignored@glossaries
1562 {%
1563 \protected@edef\@ignored@glossaries{#1}%
1564 }%
1565 {%
1566 \protected@eappto\@ignored@glossaries{,#1}%
1567 }%
1568 \csgdef{glolist@#1}{,}%
1569 \ifcsundef{gls@#1@entryfmt}%
1570 {%

```

```

1571     \defglentryfmt[#1]{\glentryfmt}%
1572   }%
1573   {}%
1574 }%
1575 }

```

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

```

1576 \newcommand*\glxtrcopytoglossary[2]{%
1577   \glstoifexists{#1}%
1578   {%
1579     \ifcsdef{glolist@#2}
1580     {%
1581       \cseappto{glolist@#2}{#1,}%
1582     }%
1583     {%
1584       \glxtrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
1585     }%
1586   }%
1587 }

```

1.3.1 Existence Checks

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

```

1588 \renewcommand{\glstoifexists}[2]{%
1589   \ifglentryexists{#1}{#2}%
1590   {%

```

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

```

1591     \protected@edef\glslabel{\glstetoklabel{#1}}%
1592     \glxtrundefaction{Glossary entry ‘\glslabel’
1593       has not been defined}{You need to define a glossary entry before
1594       you can reference it.}%
1595   }%
1596 }

```

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

```

1597 \renewcommand{\glstoifnoexists}[2]{%
1598   \ifglentryexists{#1}{%
1599     \glxtrundefaction{Glossary entry ‘\glstetoklabel{#1}’
1600       has already been defined}{}}{#2}%
1601 }

```

`\glstoifexistsordo` Modify `\glstoifexistsordo` to take account of the undefaction setting. This command was introduced in glossaries version 4.19, so check if it has been defined first.

```

1602 \ifdef\glstoifexistsordo
1603 {%

```

```

1604 \renewcommand{\glsdoifexistsordo}[3]{%
1605   \ifglstryexists{#1}{#2}%
1606   {%
1607     \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
1608       has not been defined}{You need to define a glossary entry
1609       before you can use it.}%
1610     #3%
1611   }%
1612 }%
1613 }
1614 {%
1615   \glstr@warnonexistsordo\glsdoifexistsordo
1616   \newcommand{\glsdoifexistsordo}[3]{%
1617     \ifglstryexists{#1}{#2}%
1618     {%
1619       \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
1620         has not been defined}{You need to define a glossary entry
1621         before you can use it.}%
1622       #3%
1623     }%
1624   }%
1625 }

```

arynoexistsordo Similarly for \doifglossarynoexistsordo.

```

1626 \ifdef\doifglossarynoexistsordo
1627 {%
1628   \renewcommand{\doifglossarynoexistsordo}[3]{%
1629     \ifglossaryexists*{#1}%
1630     {%
1631       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1632       #3%
1633     }%
1634     {#2}%
1635   }%
1636 }
1637 {%
1638   \glstr@warnonexistsordo\doifglossarynoexistsordo
1639   \newcommand{\doifglossarynoexistsordo}[3]{%
1640     \ifglossaryexists*{#1}%
1641     {%
1642       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1643       #3%
1644     }%
1645     {#2}%
1646   }%
1647 }
1648

```

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).

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

```
1649 \appto\@newglossaryentryposthook{%
1650   \ifdefvoid\@glo@see
1651     {\csxdef{glo@\@glo@label @see}{}}%
1652     {%
1653       \csxdef{glo@\@glo@label @see}{\@glo@see}%
1654       \if@glxtr@autoseeindex
1655         \@glxtr@autoindexcrossrefs
1656       \fi
1657     }%
1658 }
1659 \appto\@gls@keymap{,{see}{see}}
```

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

```
1660 \newcommand*{\glxtrusesee}[1]{%
1661   \glsdoifexists{#1}%
1662   {%
1663     \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@see}%
1664     \ifdefempty\@glo@see
1665       {}%
1666       {%
1667         \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
1668       }%
1669     }%
1670 }
```

`\glxtr@usesee`

```
1671 \newcommand*{\glxtr@usesee}[1][\seename]{%
1672   \@glxtr@usesee[#1]%
1673 }
```

`\@glxtr@usesee`

```
1674 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
1675   \glxtruseseeformat{#1}{#2}%
1676 }
```

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

```
1677 \newcommand*{\glxtruseseeformat}[2]{%
1678   \glsseeformat{#1}{#2}%
1679 }
```

`\glsseeitemformat` 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.)

```
1680 \renewcommand*{\glsseeitemformat}[1]{%
1681   \ifglshasshort{#1}{\glsfmttext{#1}}{\glsfmtname{#1}}%
1682 }
```

`\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.

```
1683 \newcommand*{\glsxtrhiername}[1]{%
1684   \glsdoifexists{#1}%
1685   {%
1686     \glsxtrifhasfield{parent}{#1}%
1687     {\glsxtrhiername{\glscurrentfieldvalue}\glsxtrhiernamesep}%
1688     }%
1689   \ifglshasshort{#1}{\glsfmttext{#1}}{\glsfmtname{#1}}%
1690   }%
1691 }
```

`\Glsxtrhiername`

`\Glsxtrhiername{<label>}`

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

```
1692 \newcommand*{\Glsxtrhiername}[1]{%
1693   \glsdoifexists{#1}%
1694   {%
1695     \glsxtrifhasfield{parent}{#1}%
1696     {%
1697       \Glsxtrhiername{\glscurrentfieldvalue}\glsxtrhiernamesep
1698       \ifglshasshort{#1}{\glsfmttext{#1}}{\glsfmtname{#1}}%
1699     }%
1700     {\ifglshasshort{#1}{\Glsfmttext{#1}}{\Glsfmtname{#1}}}%
1701   }%
1702 }
```

`\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.)

```
1703 \newcommand*{\GlsXtrhiername}[1]{%
1704   \glsdoifexists{#1}%
1705   {%
1706     \glstrifhasfield{parent}{#1}%
1707     {\GlsXtrhiername{\glscurrentfieldvalue}\glstrhiernamesep}%
1708     }%
1709   \ifglshasshort{#1}{\Glsfmttext{#1}}{\Glsfmtname{#1}}%
1710   }%
1711 }
```

`\GLSxtrhiername`

`\GLSxtrhiername{<label>}`

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

```
1712 \newcommand*{\GLSxtrhiername}[1]{%
1713   \glsdoifexists{#1}%
1714   {%
1715     \glstrifhasfield{parent}{#1}%
1716     {%
1717       \GLSxtrhiername{\glscurrentfieldvalue}\glstrhiernamesep
1718       \ifglshasshort{#1}{\Glsfmttext{#1}}{\Glsfmtname{#1}}%
1719     }%
1720     {\ifglshasshort{#1}{\GLSfmttext{#1}}{\GLSfmtname{#1}}}%
1721   }%
1722 }
```

`\GLSXTRhiername`

`\GLSXTRhiername{<label>}`

As above but displays all names in all-caps.

```
1723 \newcommand*{\GLSXTRhiername}[1]{%
1724   \glsdoifexists{#1}%
1725   {%
1726     \glstrifhasfield{parent}{#1}%
1727     {\GLSXTRhiername{\glscurrentfieldvalue}\glstrhiernamesep}%
1728     {}
1729   \ifglshasshort{#1}{\GLSfmttext{#1}}{\GLSfmtname{#1}}%
1730   }%
1731 }
```

`\glxtrhiernamesep` Separator used in `\glxtrhiername` and variants.

```
1732 \newcommand*{\glxtrhiernamesep}{\,\{\small$\triangleright$\}\,}
```

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

```
1733 \newcommand*{\glxtruseseealso}[1]{%
1734   \glstoifexists{#1}%
1735   {%
1736     \letcs{\@glo@see}{glo@\glstoifexists{#1}@see}%
1737     \ifdefempty\@glo@see
1738     {}%
1739     {%
1740       \expandafter\glxtruseseealsoformat\expandafter{\@glo@see}%
1741     }%
1742   }%
1743 }
```

`\glxtrusealias` Apply `\glseeformat` 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.

```
1744 \newcommand*{\glxtrusealias}[1]{%
1745   \glstoifexists{#1}%
1746   {%
1747     \letcs{\@glo@see}{glo@\glstoifexists{#1}@alias}%
1748     \ifdefempty\@glo@see
1749     {}%
1750     {%
```

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

```
1751       \glxtruseseeformat{\seename}{\@glo@see}%
1752     }%
1753   }%
1754 }
```

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

```
1755 \newcommand*{\glxtrusealsoformat}[1]{%
1756   \glseeformat[\seesalsoname]{#1}%
1757 }
```

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

```
1758 \newrobustcmd{\glxtrseelist}[1]{%
1759   \protected@edef\@glo@tmp{\noexpand\glseeelist{#1}}\@glo@tmp
1760 }
```

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

```
1761 \ifdef\alsoname
```

```

1762 {\providecommand{\seealsoname}{\alsoname}}
1763 {\providecommand{\seealsoname}{see also}}

```

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

```

1764 \ifdef\@xdycrossrefhook
1765 {

```

Add the cross-reference class definition to the hook.

```

1766 \appto\@xdycrossrefhook{%
1767 \write\glswrite{(define-crossref-class \string"seealso\string"
1768 :unverified )}%
1769 \write\glswrite{(markup-crossref-list
1770 :class \string"seealso\string"^^J\space\space\space
1771 :open \string"\string\glsxtruseseealsoformat\glsopenbrace\string"
1772 :close \string"\glsclosebrace\string")}%
1773 }

```

Append to class list.

```

1774 \appto\@xdylocationclassorder{\space\string"seealso\string"}

```

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

```

1775 \newrobustcmd*{\glsxtrindexseealso}[2]{%
1776 \ifx\@glsxtr@record@setting\@glsxtr@record@setting@alsoindex
1777 \@glsxtr@recordsee{#1}{#2}%
1778 \fi
1779 \glsdoifexists{#1}%
1780 {%
1781 \@@glsxtrwrglossmark
1782 \def\@gls@xref{#2}%
1783 \@onelevel@sanitize\@gls@xref
1784 \@gls@checkmkidxchars\@gls@xref
1785 \gls@glossary{\csname glo@#1@type\endcsname}{%
1786 (indexentry
1787 :tkey (\csname glo@#1@index\endcsname)
1788 :xref (\string"\@gls@xref\string")
1789 :attr \string"seealso\string"
1790 )
1791 }%
1792 }%
1793 }
1794 }
1795 {

```

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

```

1796 \newrobustcmd*{\glsxtrindexseealso}{\glssee[\seealsoname]}
1797 }

```

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

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

```
1798 \ifdef\gls@set@xr@key
1799 {
```

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

```
1800 \define@key{glossentry}{alias}{%
1801   \gls@set@xr@key{alias}{\@glo@alias}{#1}%
1802 }
1803 \define@key{glossentry}{seealso}{%
1804   \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
1805 }
```

Add to the key mappings.

```
1806 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}
```

Set the default value.

```
1807 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%
```

Assign the field values.

```
1808 \appto\@newglossaryentryposthook{%
1809   \ifdefvoid\@glo@seealso
1810     {\csxdef{glo@\@glo@label @seealso}{}}%
1811     {%
1812       \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1813       \if@glxtr@autoindex
1814         \@glxtr@autoindexcrossrefs
1815       \fi
1816     }%
1817 }
```

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

```
1817 \ifdefvoid\@glo@alias
1818   {\csxdef{glo@\@glo@label @alias}{}}%
1819   {%
1820     \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1821   }%
1822 }
```

Provide user-level commands to access the values.

`\glxtralias`

```
1823 \newcommand*{\glxtralias}[1]{\@gls@entry@field{#1}{alias}}
```

`trseealsolabels`

```
1824 \newcommand*{\glxtrseealsolabels}[1]{\@gls@entry@field{#1}{seealso}}
```

Add to the \@glo@autosee hook.

```

1825 \appto\@glo@autoseehook{%
1826   \ifdefvoid\@glo@alias
1827     {%
1828       \ifdefvoid\@glo@seealso
1829         {}%
1830       {%

1831         \protected@edef\@do@glsee{\noexpand\glxtrindexseealso
1832           {\@glo@label}{\@glo@seealso}}%
1833         \@do@glsee
1834       }%
1835     }%
1836   {%

```

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

```

1837   \ifdefvoid\@glo@see
1838     {%
1839       \protected@edef\@do@glsee{\noexpand\glsee{\@glo@label}{\@glo@alias}}%
1840       \@do@glsee
1841     }%
1842     {}%
1843   }%
1844 }%
1845 }
1846 {

```

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

\glxtralias

```

1847 \glsaddstoragekey*{alias}{}{\glxtralias}

```

trseealsolabels

```

1848 \glsaddstoragekey*{seealso}{}{\glxtrseealsolabels}

```

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.

```

1849 \appto\@newglossaryentryposthook{%
1850   \ifcsvoid{glo@\@glo@label @alias}%
1851     {%
1852       \ifcsvoid{glo@\@glo@label @seealso}%
1853         {}%
1854       {%

1855         \protected@edef\@do@glsee{\noexpand\glxtrindexseealso
1856           {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1857         \@do@glsee
1858       }%

```

```

1859 }%
1860 {%

```

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

```

1861 \ifdefvoid\@glo@see
1862 {%
1863     \protected@edef\@do@glsssee{\noexpand\glsssee
1864         {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1865     \@do@glsssee
1866 }%
1867 {}%
1868 }%
1869 }
1870 }

```

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

```

1871 \AtEndDocument{\if@glxtrindexcrossrefs\glxtraddallcrossrefs\fi}

```

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

```

1872 \newcommand*\glxtraddallcrossrefs{%
1873     \forallglossaries{\@glo@type}%
1874     {%
1875         \forglsentries[\@glo@type]{\@glo@label}%
1876         {%
1877             \ifglssused{\@glo@label}%
1878             {\expandafter\glxtr@addunusedxrefs\expandafter{\@glo@label}}}%
1879         }%
1880     }%
1881 }

```

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

```

1882 \newcommand*\@glxtr@addunusedxrefs[1]{%
1883     \letcs{\@glo@see}{glo\@glstetoklabel{#1}@see}%
1884     \ifdefvoid\@glo@see
1885     {}%
1886     {%
1887         \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1888     }%
1889     \letcs{\@glo@see}{glo\@glstetoklabel{#1}@seealso}%
1890     \ifdefvoid\@glo@see
1891     {}%
1892     {%
1893         \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1894     }%
1895 }

```

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

```
1896 \newcommand*{\glxtr@addunused}[1][]{%
1897   \glxtr@addunused
1898 }
```

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

```
1899 \def\glxtr@addunused#1\endglxtr@addunused{%
1900   \for\glxtr@label:=#1\do
1901   {%
1902     \ifglused{\glxtr@label}{}%
1903     {%
1904       \glsadd[format=glxtrunusedformat]{\glxtr@label}%
1905       \glsunset{\glxtr@label}%
1906       \expandafter\glxtr@addunusedxrefs\expandafter{\glxtr@label}%
1907     }%
1908   }%
1909 }
```

xtrunusedformat

```
1910 \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.

```
1911 \ifdef\gls@begindocdefs
1912 {%
1913   \renewcommand*{\gls@begindocdefs}{%
1914     \ifnum\glxtr@docdefval=1\relax
1915       \gls@enablesavenonumberlist
1916       \edef\gls@restoreat{%
1917         \noexpand\catcode'\noexpand\@=\number\catcode'\@ \relax}%
1918       \makeatletter
1919       \InputIfFileExists{\jobname.glsdefs}{\relax}%
1920       \gls@restoreat
1921       \undef\gls@restoreat
1922       \gls@defdocnewglossaryentry
1923     \else
1924       \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.

```
1925       \gls@enablesavenonumberlist
1926       \let\gls@checkseeallowed\relax
1927       \let\newglossaryentry\new@atomglossaryentry
1928       \global\newwrite\gls@deffile
1929       \immediate\openout\gls@deffile=\jobname.glsdefs
```

Write all currently defined entries.

```

1930      \forallglsentries{\@glsentry}{\@gls@writedef{\@glsentry}}%
1931      \fi
1932    \fi
1933  }
1934 }
1935 {%
1936   \ifnum\@glsxtr@docdefval=3\relax
1937     \PackageError{glossaries-extra}{Package option
1938       'docdef=\@glsxtr@docdefsetting' requires at least version 4.37
1939       of the base glossaries.sty package}{}
1940   \fi
1941 }

```

m@glossaryentry

```

1942 \newrobustcmd{\new@atom@glossaryentry}[2]{%
1943   \gls@defglossaryentry{#1}{#2}%
1944   \@gls@writedef{#1}%
1945 }

```

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.

```

1946 \let\glsxtr@orgmakenoidxglossaries\makenoidxglossaries
1947 \renewcommand{\makenoidxglossaries}{%
1948   \@domakeglossaries
1949   {%
1950     \ifdefequal\@glsxtr@record@setting\@glsxtr@record@setting@off
1951     {%
1952       \glsxtr@orgmakenoidxglossaries

```

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

```

1953   \renewcommand{\@do@seeglossary}[2]{%
1954     @@glsxtrwrglossmark

1955     \protected@edef\@gls@label{\glsdetoklabel{##1}}%
1956     \protected@write\@auxout{}{%
1957       \string\@gls@reference
1958       {\csname glo@\@gls@label @type\endcsname}%
1959       {\@gls@label}%
1960       {%
1961         \string\glsseeformat##2}%
1962       }%
1963     }%
1964   }%

```

Check for docdefs=restricted:

```

1965   \if@glsxtrdocdefrestricted

```


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

```

1966 \renewcommand*{\@gls@reference}[3]{%
1967 \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
1968 \ifinlistcs{##2}{\glsref@##1}%
1969 {}%
1970 {\listcsgadd{\glsref@##1}{##2}}%
1971 \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
1972 {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
1973 {}%
1974 \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
1975 }%
1976 \else

```

Disable document definitions.

```

1977 \@glsxtrdocdeffalse
1978 \fi
1979 \disable@keys{glossaries-extra}{docdef}%
1980 }%
1981 {%
1982 \PackageError{glossaries-extra}{\string\makenoidxglossaries\space
1983 not permitted\MessageBreak
1984 with record=\@glsxtr@record@setting\space package option}%
1985 {You may only use \string\makenoidxglossaries\space with the
1986 record=off option}%
1987 }%
1988 }%
1989 }

```

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

```

1990 \renewcommand*{\gls@defdocnewglossaryentry}{%
1991 \ifcase\@glsxtr@docdefval
1992 docdef=false:
1993 \renewcommand*{\newglossaryentry}[2]{%
1994 \PackageError{glossaries-extra}{Glossary entries must
1995 be \MessageBreak defined in the preamble with \MessageBreak
1996 package option 'docdef=false'\MessageBreak(consider using
1997 'docdef=restricted')}{Move your glossary definitions to
1998 the preamble. You can also put them in a \MessageBreak separate file
1999 and load them with \string\loadglsentries.}%
2000 }%
2001 \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.

```

2001 \let\gls@checkseeallowed\relax
2002 \let\newglossaryentry\newglossaryentry
2003 \else

```

Restricted mode just needs to allow the see value.

```
2004 \let\gls@checkseeallowed\relax
2005 \fi
2006 }%
```

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`

```
2007 \newcommand*{\GlsXtrEnableOnTheFly}{%
2008 \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
2009 }
```

`\rEnableOnTheFly`

The starred version attempts to allow UTF8 characters in the label, but this may break! (Formatting commands mustn't be used in the label, but the label may be a command whose replacement text is the actual label. This doesn't take into account a command that's defined in terms of another command that may eventually expand to the label text.)

```
2010 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
2011 \renewcommand*{\glsdetoklabel}[1]{%
2012 \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
2013 {%
2014 \expandafter\detokenize\expandafter{##1}%
2015 }%
2016 {\detokenize{##1}}}%
2017 }%
2018 \@GlsXtrEnableOnTheFly
2019 }
2020 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
2021 \expandafter\if\glsbackslash#1%
2022 #3%
2023 \else
2024 #4%
2025 \fi
2026 }
```

`\sXtrstarflywarn`

```
2027 \newcommand*{\glsxtrstarflywarn}{%
2028 \GlossariesExtraWarning{Experimental starred version of
2029 \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
2030 read the warnings in the glossaries-extra user manual)}}%
2031 }
```

`\rEnableOnTheFly`

```
2032 \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:

```

\glstrcat
2033 \newcommand*\glstrcat[1]{}

\glstr
2034 \newcommand*\glstr[1]{}
2035 \def\glstr@keylist{##1}%
2036 \@glstr
2037 }

\@glstr
2038 \newcommand*\@glstr[2]{}
2039 \ifglstryexists{##2}%
2040 {%
2041   \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
2042 }%
2043 {%
2044   \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
2045     description={\nopostdesc},##1}%
2046 }%
2047 \expandafter\gls\expandafter[\glstr@keylist]{##2}%
2048 }

\Glsxtr
2049 \newcommand*\Glsxtr[1]{}
2050 \def\Glsxtr@keylist{##1}%
2051 \@Glsxtr
2052 }

\@Glsxtr
2053 \newcommand*\@Glsxtr[2]{}
2054 \ifglstryexists{##2}%
2055 {%
2056   \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
2057 }%
2058 {%
2059   \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
2060     description={\nopostdesc},##1}%
2061 }%
2062 \expandafter\Gls\expandafter[\Glsxtr@keylist]{##2}%
2063 }

\glstrpl
2064 \newcommand*\glstrpl[1]{}
2065 \def\glstrpl@keylist{##1}%
2066 \@glstrpl
2067 }

\@glstrpl

```

```

2068 \newcommand*{\@glxstrpl}[2][{}]{%
2069 \ifglxentryexists{##2}%
2070 {%
2071 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
2072 }%
2073 {%
2074 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
2075 description={\nopostdesc},##1}%
2076 }%
2077 \expandafter\glsp\expandafter[\glxtr@keylist]{##2}%
2078 }

```

\Glsxtrpl

```

2079 \newcommand*{\Glsxtrpl}[1][{}]{%
2080 \def\glxtr@keylist{##1}%
2081 \@Glsxtrpl
2082 }

```

\@Glsxtrpl

```

2083 \newcommand*{\@Glsxtrpl}[2][{}]{%
2084 \ifglxentryexists{##2}
2085 {%
2086 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
2087 }%
2088 {%
2089 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
2090 description={\nopostdesc},##1}%
2091 }%
2092 \expandafter\glsp\expandafter[\glxtr@keylist]{##2}%
2093 }

```

\GlsXtrWarning

```

2094 \newcommand*{\GlsXtrWarning}[2]{%
2095 \def\glxtr@optlist{##1}%
2096 \@onelevel@sanitize\glxtr@optlist
2097 \GlossariesExtraWarning{The options ‘\glxtr@optlist’ have
2098 been ignored for entry ‘##2’ as it has already been defined}%
2099 }

```

Disable commands after the glossary:

```

2100 \renewcommand\@printglossary[2]{%
2101 \def\glxtr@printglossopts{##1}%
2102 \@glxtr@orgprintglossary{##1}{##2}%
2103 \def\glxtr{\@glxtr@disabledflycommand\glxtr}%
2104 \def\glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
2105 \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
2106 \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
2107 }

```

abledflycommand

```

2108 \newcommand*{\@glxtr@disabledflycommand}[1]{%
2109   \PackageError{glossaries-extra}%
2110   {\string##1\space can't be used after any of the \MessageBreak
2111    glossaries have been displayed}%
2112   {The on-the-fly commands enabled by
2113    \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
2114    before the glossaries. If you want to use any entries \MessageBreak
2115    after any of the glossaries, you must use the standard \MessageBreak
2116    method of first defining the entry and then using the \MessageBreak
2117    entry with commands like \string\gls}%
2118   \@glxtr@disabledflycommand
2119 }%
2120 \newcommand*{\@glxtr@disabledflycommand}[2][\{##2}

```

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

```

2121 \let\GlsXtrEnableOnTheFly\relax
2122 }
2123 \@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.

```

2124 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}

```

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

etglossarystyle

```

2125 \renewcommand*{\setglossarystyle}[1]{%
2126   \ifcsundef{@glsstyle@#1}%
2127   {%
2128     \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%
2129   }%
2130   {%
2131     \csname @glsstyle@#1\endcsname

```

Only set the current style if it exists.

```

2132   \protected@edef\@glxtr@current@style{#1}%
2133 }%
2134 \ifx\@glossary@default@style\relax
2135   \protected@edef\@glossary@default@style{#1}%
2136 \fi
2137 }

```

In case we have an old version of glossaries:

```

2138 \ifdef\@glossary@default@style
2139 {}

```

```

2140 {%
2141   \let@glossary@default@style\relax
2142 }

```

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

```

2143 \ifdef\glslistdottedwidth
2144 {%
2145   \ifdim\glslistdottedwidth=.5\hsize
2146     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
2147     \AtBeginDocument{%
2148       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
2149         \setlength{\glslistdottedwidth}{.5\columnwidth}%
2150       \fi
2151     }%
2152   \fi
2153 }
2154 {}%

```

Similarly for \glsdescwidth:

\glsdescwidth

```

2155 \ifdef\glsdescwidth
2156 {%
2157   \ifdim\glsdescwidth=.6\hsize
2158     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
2159     \AtBeginDocument{%
2160       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
2161         \setlength{\glsdescwidth}{.6\columnwidth}%
2162       \fi
2163     }%
2164   \fi
2165 }
2166 {}%

```

and for \glspagelistwidth:

lspagelistwidth

```

2167 \ifdef\glspagelistwidth
2168 {%
2169   \ifdim\glspagelistwidth=.1\hsize
2170     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
2171     \AtBeginDocument{%
2172       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
2173         \setlength{\glspagelistwidth}{.1\columnwidth}%
2174       \fi
2175     }%
2176   \fi
2177 }
2178 {}%

```

aryentrynumbers Has the nonumberlist option been used?

```
2179 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
2180 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
2181   \glsnonumberlistfalse
2182   \renewcommand*{\glossaryentrynumbers}[1]{%
2183     \ifglsentryexists{\glscurrententrylabel}%
2184     {%
2185       \@glsxtrpreloctag
2186       \GlsXtrFormatLocationList{#1}%
2187       \@glsxtrpostloctag
2188       \gls@save@numberlist{#1}%
2189     }{}%
2190   }%
2191 \else
2192   \glsnonumberlisttrue
2193   \renewcommand*{\glossaryentrynumbers}[1]{%
2194     \ifglsentryexists{\glscurrententrylabel}%
2195     {%
2196       \gls@save@numberlist{#1}%
2197     }{}%
2198   }%
2199 \fi
```

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

```
2200 \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

```
2201 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
2202   \let\@glsxtrpreloctag\@glsxtrpreloctag
2203   \let\@glsxtrpostloctag\@glsxtrpostloctag
2204   \renewcommand*{\@glsxtr@pagetag}{#1}%
2205   \renewcommand*{\@glsxtr@pagestag}{#2}%
2206   \renewcommand*{\@glsxtr@savepreloctag}[2]{%
2207     \csgdef{\@glsxtr@preloctag@##1}{##2}%
2208   }%
2209   \renewcommand*{\@glsxtr@doloctag}{%
2210     \ifcsundef{\@glsxtr@preloctag\@glscurrententrylabel}%
2211     {%
2212       \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
2213         Rerun required}%
2214     }%
2215   }%
```

```

2216      \csuse{@glxstr@preloctag@glscurrententrylabel}%
2217    }%
2218  }%
2219 }
2220 \@onlypreamble\GlsXtrEnablePreLocationTag

```

glxstrpreloctag

```

2221 \newcommand*{\@@glxstrpreloctag}{%
2222   \let\@glxstr@org@delimN\delimN
2223   \let\@glxstr@org@delimR\delimR
2224   \let\@glxstr@org@glsgignore\glsgignore
2225   \gdef is required as the delimiters may occur inside a scope.
2226   \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
2227   \renewcommand*{\delimN}{%
2228     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
2229     \@glxstr@org@delimN}%
2230   \renewcommand*{\delimR}{%
2231     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
2232     \@glxstr@org@delimR}%
2233   \renewcommand*{\glsgignore}[1]{%
2234     \gdef\@glxstr@thisloctag{\relax}%
2235     \@glxstr@org@glsgignore{##1}}%
2236   \@glxstr@doloctag
2237 }

```

glxstrpreloctag

```

2237 \newcommand*{\@glxstrpreloctag}{%

```

@glxstr@pagetag

```

2238 \newcommand*{\@glxstr@pagetag}{%

```

glxstr@pagetag

```

2239 \newcommand*{\@glxstr@pagetag}{%

```

lsxtrpostloctag

```

2240 \newcommand*{\@@glxtrpostloctag}{%
2241   \let\delimN\@glxtr@org@delimN
2242   \let\delimR\@glxtr@org@delimR
2243   \let\glsgignore\@glxtr@org@glsgignore
2244   \protected@write\@auxout{%
2245     {\string\@glxtr@savepreloctag\glscurrententrylabel}\@glxtr@thisloctag}}%
2246 }

```

lsxtrpostloctag

```

2247 \newcommand*{\@glxtrpostloctag}{%

```


lsxtr@preloctag

```
2248 \newcommand*{\@glxtr@savepreloctag}[2]{%
2249 \protected@write\@auxout{}{%
2250   \string\providecommand\string\@glxtr@savepreloctag[2]{}}
```

glxtr@doloctag

```
2251 \newcommand*{\@glxtr@doloctag}{}
```

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

```
2252 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
2253   \XKV@plfalse
2254   \XKV@sttrue
2255   \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
2256   {%
2257     \csname glsnonumberlist\XKV@resa\endcsname
2258     \ifglsnonumberlist
2259       \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
2260     \else
2261       \def\glossaryentrynumbers##1{%
2262         \@glxtr@preloctag
2263         \GlsXtrFormatLocationList{##1}%
2264         \@glxtr@postloctag
2265         \gls@save@numberlist{##1}}%
2266       \fi
2267   }%
2268 }
```

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.

```
2269 \renewcommand*{\glsentryfmt}{%
2270   \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}{}}%
2271   \glsifregular{\glslabel}%
2272   {\glxtrregularfont{\glsentryfmt}}%
2273   {%
2274     \ifglshasshort{\glslabel}%
2275     {\glxtrabbreviationfont{\glxtrgenabbrvfmt}}%
2276     {\glxtrregularfont{\glsentryfmt}}%
2277   }%
2278 }
```

`sxtrregularfont` Font used for regular entries.

```
2279 \newcommand*{\glsxtrregularfont}[1]{#1}
```

`bbreviationfont` Font used for abbreviation entries.

```
2280 \newcommand*{\glxtrabbreviationfont}[1]{#1}
```

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

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

```
2281 \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).

```
2282 \@glxtr@record{#2}{#3}{glslink}%
```

```
2283 \glsdoifexists{#3}%
```

```
2284 {%
```

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).

```
2285 \let\glxtrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
```

```
2286 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
2287 \def\glscustomtext{#4}%
```

```
2288 \@glxtr@field@linkdefs
```

```
2289 #1%
```

```
2290 \@gls@link[#2]{#3}{#4}%
```

```
2291 \let\ifKV@glslink@hyper\glxtrorg@ifKV@glslink@hyper
```

```
2292 }%
```

```
2293 \glspostlinkhook
```

```
2294 }
```

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

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

```
2295 \let\@glxtr@org@gls@\@gls@
```

```
2296 \def\@gls@#1#2{%
```

```
2297 \@glxtr@record{#1}{#2}{glslink}%
```

```
2298 \@glxtr@org@gls@{#1}{#2}%
```

```
2299 }%
```

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

```
2300 \let\@glxtr@org@glspl@\@glspl@
```

```
2301 \def\@glspl@#1#2{%
```

```

2302 \@glstr@record{#1}{#2}{glslink}%
2303 \@glstr@org@glsp1@{#1}{#2}%
2304 }%

```

\@Gls@ Save the original definition and redefine.

```

2305 \let\@glstr@org@Gls@\@Gls@
2306 \def\@Gls@#1#2{%
2307 \@glstr@record{#1}{#2}{glslink}%
2308 \@glstr@org@Gls@{#1}{#2}%
2309 }%

```

\@Glspl@ Save the original definition and redefine.

```

2310 \let\@glstr@org@Glspl@\@Glspl@
2311 \def\@Glspl@#1#2{%
2312 \@glstr@record{#1}{#2}{glslink}%
2313 \@glstr@org@Glspl@{#1}{#2}%
2314 }%

```

\@GLS@ Save the original definition and redefine.

```

2315 \let\@glstr@org@GLS@\@GLS@
2316 \def\@GLS@#1#2{%
2317 \@glstr@record{#1}{#2}{glslink}%
2318 \@glstr@org@GLS@{#1}{#2}%
2319 }%

```

\@GLSpl@ Save the original definition and redefine.

```

2320 \let\@glstr@org@GLSpl@\@GLSpl@
2321 \def\@GLSpl@#1#2{%
2322 \@glstr@record{#1}{#2}{glslink}%
2323 \@glstr@org@GLSpl@{#1}{#2}%
2324 }%

```

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

```

2325 \renewcommand*{\@glsdisp}[3][{}]{%
2326 \@glstr@record{#1}{#2}{glslink}%
2327 \glsdoifexists{#2}{%
2328 \let\do@gl@link@checkfirsthyper\@gl@link@checkfirsthyper
2329 \let\glsifplural\@secondoftwo
2330 \let\glscapscase\@firstofthree
2331 \def\glscustomtext{#3}%
2332 \def\glsinsert{}%
2333 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
2334 \@gl@link[#1]{#2}{\@glo@text}%
2335 \ifKV@glslink@local
2336 \glslocalunset{#2}%
2337 \else
2338 \glsunset{#2}%

```

```

2339     \fi
2340 }%
2341 \glspostlinkhook
2342 }

```

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

```

2343 \renewcommand*{\@gls@link}[3][\%
2344   \@glxtr@record{#1}{#2}{glslink}%
2345   \glsdoifexistsordo{#2}%
2346   {%
2347     \let\do@gls@link@checkfirsthyper\relax
2348     Post-link hook commands need initialising.
2349     \def\glscustomtext{#3}%
2350     \@glxtr@field@linkdefs
2351     \@gls@link[#1]{#2}{#3}%
2352   }%
2353   \glstextformat{#3}%
2354 }%
2355 \glspostlinkhook
2356 }

```

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

```

2357 \newcommand*{\glxtrinitwrgloss}{%
2358   \glsifattribute{\glslabel}{wrgloss}{after}%
2359   {%
2360     \glxtrinitwrglossbeforefalse
2361   }%
2362   {%
2363     \glxtrinitwrglossbeforetrue
2364   }%
2365 }

```

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

```

2366 \newif\ifglxtrinitwrglossbefore
2367 \glxtrinitwrglossbeforetrue

```

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

```

2368 \define@choicekey{glslink}{wrgloss}%
2369 [\@glxtr@wrglossval\@glxtr@wrglossnr]%
2370 {before,after}%
2371 {%
2372   \ifcase\@glxtr@wrglossnr\relax
2373   \glxtrinitwrglossbeforetrue
2374   \or
2375   \glxtrinitwrglossbeforefalse
2376   \fi
2377 }

```

```

2378 \define@key{glslink}{thevalue}{\def\@glstr@thevalue{#1}}

2379 \define@key{glslink}{theHvalue}{\def\@glstr@theHvalue{#1}}

tr@hyperoutside  Define a hyperoutside key to determine whether \hyperlink should be outside \glstextformat.
2380 \define@boolkey{glslink}[glstr@]{hyperoutside}[true]{}
2381 \glstr@hyperoutsidettrue

ocal@textformat  Provide a key to locally change the text format.
2382 \define@key{glslink}{textformat}{%
2383   \ifcsdef{#1}
2384   {%
2385     \letcs{\@glstr@local@textformat}{#1}%
2386   }%
2387   {%
2388     \PackageError{glossaries-extra}{Unknown control sequence name ‘#1’}{}%
2389   }%
2390 }

2391 \define@key{glslink}{prefix}{\def\glolinkprefix{#1}}

nithyperoutside  Set the default if the hyperoutside is omitted.
2392 \newcommand*{\glstr@nithyperoutside}{%
2393   \glsifattribute{\glslabel}{hyperoutside}{false}%
2394   {%
2395     \glstr@hyperoutsidetfalse
2396   }%
2397   {%
2398     \glstr@hyperoutsidettrue
2399   }%
2400 }

r@inc@linkcount  Does nothing by default.
2401 \newcommand*{\glstr@inc@linkcount}{}

slinkpresetkeys  User hook performed immediately before options are set. Does nothing by default.
2402 \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.
2403 \newrobustcmd*{\GlsXtrExpandedFmt}[2]{%
2404   \protected@edef\@glstr@tmp{#2}%
2405   \expandafter#1\expandafter{\@glstr@tmp}%
2406 }

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.
2407 \newcommand*{\@glstr@use@equation@counter}{%
2408   \@glstr@ifnum@mmode{\def\@gls@counter{equation}}{}%
2409 }

```

`\glstr@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 argument indicates the key family (`glslink` or `glossadd`).

```
2410 \newcommand*{\glstr@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.

```
2411 \newcommand*{\GlsXtrAutoAddOnFormat}[3][\glslabel]{%
2412   \renewcommand*{\glstr@do@autoadd}[1]{%
2413     \begingroup
2414       \protected@edef\glstr@do@autoadd{%
2415         \noexpand\ifstrequal{##1}{\glslink}%
2416         {%
2417           \noexpand\DTLifinlist{\@glsnumberformat}{#2}{\noexpand\glsadd[format={\@glsnumberfor
2418           }%
2419           }%
2420         }%
2421       \glstr@do@autoadd
2422     \endgroup
2423   }%
2424 }
```

`\@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`.

```
2425 \def\@gls@link[#1]#2#3{%
2426   \leavevmode

2427   \protected@edef\glslabel{\glsdetoklabel{#2}}%
2428   \def\@gls@link@opts{#1}%
2429   \let\@gls@link@label\glslabel
2430   \let\@glsnumberformat\glstr@defaultnumberformat
2431   \protected@edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
2432   \protected@edef\gls@type{\csname glo@\glslabel @type\endcsname}%
2433   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
```

Save current value of `\glolinkprefix`:

```
2434 \let\@glstr@org@glolinkprefix\glolinkprefix
```

Initialise `\@glstr@local@textformat`

```
2435 \let\@glstr@local@textformat\relax
```

Initialise thevalue and theHvalue (v1.19).

```

2436 \def\@glxstr@thevalue{}%
2437 \def\@glxstr@theHvalue{\@glxstr@thevalue}%

```

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

```

2438 \glxstrinitwrgloss

```

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

```

2439 \glxstrinithyperoutside

```

Note that the default link options may override \glxstrinitwrgloss.

```

2440 \@gls@setdefault@glslink@opts

```

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

```

2441 \glxstr@inc@linkcount

```

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

```

2442 \if@glxstr@equations
2443     \@glxstr@use@equation@counter
2444 \fi

```

As the original definition.

```

2445 \do@gl:disablehyperinlist
2446 \do@gl:link@checkfirsthyper

```

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

```

2447 \glslinkpresetkeys

```

Set options.

```

2448 \setkeys{glslink}{#1}%

```

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

```

2449 \glxstr@do@autoadd{glslink}%

```

User hook after options are set:

```

2450 \glslinkpostsetkeys

```

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

```

2451 \ifdefempty{\@glxstr@thevalue}%
2452 {%
2453     \@gls@saveentrycounter
2454 }%
2455 {%
2456     \let\theglentrycounter\@glxstr@thevalue
2457     \def\theHglentrycounter{\@glxstr@theHvalue}%
2458 }%
2459 \@gls@setsort{glslabel}%

```

Check if the textformat key has been used.

```

2460 \ifx\@glxstr@local@textformat\relax

```

Check textformat attribute (new to v1.21).

```

2461     \glshasattribute{glslabel}{textformat}%
2462     {%

```

```

2463     \protected@edef\@glxstr@attrval{\glsgetattribute{\glslabel}{textformat}}%
2464     \ifcsdef{\@glxstr@attrval}%
2465     {%
2466         \letcs{\@glxstr@textformat}{\@glxstr@attrval}%
2467     }%
2468     {%
2469         \GlossariesExtraWarning{Unknown control sequence name
2470         '\@glxstr@attrval' supplied in textformat attribute
2471         for entry '\glslabel'. Reverting to default \string\glstextformat}%
2472         \let\@glxstr@textformat\glstextformat
2473     }%
2474 }%
2475 {%
2476     \let\@glxstr@textformat\glstextformat
2477 }%
2478 \else
2479     \let\@glxstr@textformat\@glxstr@local@textformat
2480 \fi

```

Do write if it should occur before the link text:

```

2481 \ifglxstrinitwrglossbefore
2482 \do@wrglossary{#2}%
2483 \fi

```

Do the link text:

```

2484 \ifKV@glslink@hyper
2485     \ifglxstr@hyperoutside
2486         \@glslink{\glolinkprefix\glslabel}{\@glxstr@textformat{#3}}%
2487     \else
2488         \@glxstr@textformat{\@glslink{\glolinkprefix\glslabel}{#3}}%
2489     \fi
2490 \else
2491     \ifglxstr@hyperoutside
2492         \glndonohyperlink{\glolinkprefix\glslabel}{\@glxstr@textformat{#3}}%
2493     \else
2494         \@glxstr@textformat{\glndonohyperlink{\glolinkprefix\glslabel}{#3}}%
2495     \fi
2496 \fi

```

Do write if it should occur after the link text:

```

2497 \ifglxstrinitwrglossbefore
2498 \else
2499     \do@wrglossary{#2}%
2500 \fi

```

Restore original value of \glolinkprefix:

```

2501 \let\glolinkprefix\@glxstr@org@glolinkprefix

```

As the original definition:

```

2502 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
2503 }

```



```

2504 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}

2505 \define@key{glossadd}{theHvalue}{\def\@glxtr@theHvalue{#1}}

\lsaddpresetkeys
2506 \newcommand*{\@glssaddpresetkeys}{%

\lsaddpostsetkeys
2507 \newcommand*{\@glssaddpostsetkeys}{%

\glsadd   Redefine to include \@glxtr@record and suppress in headings
2508 \renewrobustcmd*{\glsadd}[2][\@glxtr@record]{%
2509   \glxtrifinmark
2510   }{%
2511   {%
2512     \@glxtr@adjustmode
2513     \begin@group
2514     \@glxtr@record{#1}{#2}{\glsadd}%
2515     \glsdoifexists{#2}%
2516     {%
2517       \let\@glxtr@numberformat\@glxtr@defaultnumberformat

2518       \protected@edef\@glxtr@counter{\csname glo@glxtr@detoklabel{#2}@counter\endcsname}%
2519       \def\@glxtr@thevalue{}%
2520       \def\@glxtr@theHvalue{\@glxtr@thevalue}%

      Implement any default settings (before options are set)
2521       \@glssaddpresetkeys
2522       \setkeys{glossadd}{#1}%

      Implement any default settings (after options are set)
2523       \@glssaddpostsetkeys
2524       \ifdefempty{\@glxtr@thevalue}%
2525       {%
2526         \@glxtr@saveentrycounter
2527       }%
2528       {%
2529         \let\theglentrycounter\@glxtr@thevalue
2530         \def\theHglentrycounter{\@glxtr@theHvalue}%
2531       }%

      Define sort key if necessary (in case of sort=use):
2532       \@glxtr@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 \glignore can
      be used for selection without location, but the indexing still needs to be performed.
2533       \KV@glxtr@noindexfalse
2534       \@do@wrglossary{#2}%
2535     }%

```

```

2536     \endgroup
2537 }%
2538 }

```

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

```

2539 \newrobustcmd{\glsaddeach}[2] []{%
2540   \for\@gls@thislabel:=#2\do{\glsadd[#1]{\@gls@thislabel}}%
2541 }

```

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

```

2542 \newcommand*{\@glsxtr@field@linkdefs}{%
2543   \let\glsxtrifwasfirstuse\@secondoftwo
2544   \let\glsifplural\@secondoftwo
2545   \let\glscapscase\@firstofthree
2546   \let\glsinsert\@empty
2547 }

```

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

`assignfieldfont`

```

2548 \newcommand*{\glsxtrassignfieldfont}[1]{%
2549   \ifglstryexists{#1}%
2550   {%
2551     \ifglshasshort{#1}%
2552     {%
2553       \glssetabbrvfmt{\glscategory{#1}}%
2554       \glsifregular{#1}%
2555       {\let\@gls@field@font\glsxtrregularfont}%
2556       {\let\@gls@field@font\@firstofone}%
2557     }%
2558     {%
2559       \glsifnotregular{#1}%
2560       {\let\@gls@field@font\@firstofone}%
2561       {\let\@gls@field@font\glsxtrregularfont}%
2562     }%
2563   }%
2564   {%
2565     \let\@gls@field@font\@gobble
2566   }%
2567 }

```

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

```

2568 \def\@glstext@#1#2[#3]{%
2569   \glsxtrassignfieldfont{#2}%
2570   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
2571 }

```

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

```
2572 \def\@GLStext@#1#2[#3]{%
2573   \glstrassignfieldfont{#2}%
2574   \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
2575   {\@gls@field@font{\Glsaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
2576 }
```

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

```
2577 \def\@Glstext@#1#2[#3]{%
2578   \glstrassignfieldfont{#2}%
2579   \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
2580   {\@gls@field@font{\Glsaccesstext{#2}#3}}}%
2581 }
```

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`

```
2582 \newcommand*{\glstrchecknohyperfirst}[1]{%
2583   \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
2584 }
```

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

```
2585 \def\@glsfirst@#1#2[#3]{%
2586   \glstrassignfieldfont{#2}%
2587   \@gls@field@link
2588   [\let\glstrifwasfirstuse\@firstoftwo
2589   \glstrchecknohyperfirst{#2}%
2590   ]{#1}{#2}%
2591   {\@gls@field@font{\Glsaccessfirst{#2}#3}}}%
2592 }
```

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

```
2593 \def\@Glsfirst@#1#2[#3]{%
2594   \glstrassignfieldfont{#2}%
2595   \@gls@field@link
2596   [\let\glstrifwasfirstuse\@firstoftwo
2597   \let\glscapscase\@secondofthree
2598   \glstrchecknohyperfirst{#2}%
2599   ]%
2600   {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}}%
2601 }
```

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

```
2602 \def\@GLSfirst@#1#2[#3]{%
2603   \glstrassignfieldfont{#2}%
2604 }
```

Ensure that \GLSfirst honours the nohyperfirst attribute.

```
2604 \@gls@field@link
2605 [\let\glstrifwasfirstuse\@firstoftwo
2606 \let\glscapscase\@thirdofthree
2607 \glstrchecknohyperfirst{#2}%
2608 ]%
2609 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
2610 }
```

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

```
2611 \def\@glsplural@#1#2[#3]{%
2612 \glstrassignfieldfont{#2}%
2613 \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
2614 {\@gls@field@font{\glsaccessplural{#2}#3}}%
2615 }
```

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

```
2616 \def\@Glsplural@#1#2[#3]{%
2617 \glstrassignfieldfont{#2}%
2618 \@gls@field@link
2619 [\let\glsifplural\@firstoftwo
2620 \let\glscapscase\@secondofthree
2621 ]%
2622 {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
2623 }
```

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

```
2624 \def\@GLSplural@#1#2[#3]{%
2625 \glstrassignfieldfont{#2}%
2626 \@gls@field@link
2627 [\let\glsifplural\@firstoftwo
2628 \let\glscapscase\@thirdofthree
2629 ]%
2630 {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
2631 }
```

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

```
2632 \def\@glsfirstplural@#1#2[#3]{%
2633 \glstrassignfieldfont{#2}%
2634 \@gls@field@link
2635 [\let\glstrifwasfirstuse\@firstoftwo
2636 \let\glsifplural\@firstoftwo
2637 \glstrchecknohyperfirst{#2}%
2638 ]%
2639 {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
2640 }
```

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

```
2641 \def\@Glsfirstplural@#1#2[#3]{%
2642   \glstrassignfieldfont{#2}%
      Ensure that \glfirstplural honours the nohyperfirst attribute.
2643   \@gls@field@link
2644   [\let\glstrifwasfirstuse\@firstoftwo
2645    \let\glstifplural\@firstoftwo
2646    \let\glscapscase\@secondofthree
2647    \glstrchecknohyperfirst{#2}%
2648   ]%
2649   {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
2650 }
```

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

```
2651 \def\@GLSfirstplural@#1#2[#3]{%
2652   \glstrassignfieldfont{#2}%
      Ensure that \glfirstplural honours the nohyperfirst attribute.
2653   \@gls@field@link
2654   [\let\glstrifwasfirstuse\@firstoftwo
2655    \let\glstifplural\@firstoftwo
2656    \let\glscapscase\@thirdofthree
2657    \glstrchecknohyperfirst{#2}%
2658   ]%
2659   {#1}{#2}%
2660   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
2661 }
```

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

```
2662 \def\@glsname@#1#2[#3]{%
2663   \glstrassignfieldfont{#2}%
2664   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
2665 }
```

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

```
2666 \def\@Glsname@#1#2[#3]{%
2667   \glstrassignfieldfont{#2}%
2668   \@gls@field@link
2669   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2670   {\@gls@field@font{\Glsaccessname{#2}#3}}%
2671 }
```

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

```
2672 \def\@GLSname@#1#2[#3]{%
2673   \glstrassignfieldfont{#2}%
2674   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2675   {#1}{#2}%
2676   {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
2677 }
```

```

\@Glsdesc@
2678 \def\@Glsdesc@#1#2[#3]{%
2679   \glstrassignfieldfont{#2}%
2680   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
2681 }

\@GLSdesc@   First letter uppercase version.
2682 \def\@GLSdesc@#1#2[#3]{%
2683   \glstrassignfieldfont{#2}%
2684   \@gls@field@link
2685   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2686   {\@gls@field@font{\GLSaccessdesc{#2}#3}}%
2687 }

\@GLSdesc@   All uppercase version.
2688 \def\@GLSdesc@#1#2[#3]{%
2689   \glstrassignfieldfont{#2}%
2690   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2691   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2692 }

@Glsdescplural@   No case-changing version.
2693 \def\@Glsdescplural@#1#2[#3]{%
2694   \glstrassignfieldfont{#2}%
2695   \@gls@field@link
2696   [\let\glscapscase\@secondoftwo
2697   \let\glsifplural\@firstoftwo
2698   ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
2699 }

@GLSdescplural@   First letter uppercase version.
2700 \def\@GLSdescplural@#1#2[#3]{%
2701   \glstrassignfieldfont{#2}%
2702   \@gls@field@link
2703   [\let\glscapscase\@secondoftwo
2704   \let\glsifplural\@firstoftwo
2705   ]{#1}{#2}{\@gls@field@font{\GLSaccessdescplural{#2}#3}}%
2706 }

@GLSdescplural@   All uppercase version.
2707 \def\@GLSdesc@#1#2[#3]{%
2708   \glstrassignfieldfont{#2}%
2709   \@gls@field@link
2710   [\let\glscapscase\@thirdoftwo
2711   \let\glsifplural\@firstoftwo
2712   ]%
2713   {#1}{#2}%
2714   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
2715 }

```

\@glssymbol@

```
2716 \def\@glssymbol@#1#2[#3]{%
2717   \glstrassignfieldfont{#2}%
2718   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
2719 }
```

\@Glsymbol@ First letter uppercase version.

```
2720 \def\@Glsymbol@#1#2[#3]{%
2721   \glstrassignfieldfont{#2}%
2722   \@gls@field@link
2723   [\let\glscapscase\@secondoftwo]%
2724   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
2725 }
```

\@GLSsymbol@ All uppercase version.

```
2726 \def\@GLSsymbol@#1#2[#3]{%
2727   \glstrassignfieldfont{#2}%
2728   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2729   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
2730 }
```

lssymbolplural@ No case-changing version.

```
2731 \def\lssymbolplural@#1#2[#3]{%
2732   \glstrassignfieldfont{#2}%
2733   \@gls@field@link
2734   [\let\glscapscase\@secondoftwo
2735   \let\glsifplural\@firstoftwo
2736   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2737 }
```

lssymbolplural@ First letter uppercase version.

```
2738 \def\lssymbolplural@#1#2[#3]{%
2739   \glstrassignfieldfont{#2}%
2740   \@gls@field@link
2741   [\let\glscapscase\@secondoftwo
2742   \let\glsifplural\@firstoftwo
2743   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2744 }
```

LSsymbolplural@ All uppercase version.

```
2745 \def\LSsymbol@#1#2[#3]{%
2746   \glstrassignfieldfont{#2}%
2747   \@gls@field@link
2748   [\let\glscapscase\@thirdoftwo
2749   \let\glsifplural\@firstoftwo
2750   ]%
2751   {#1}{#2}%
2752   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2753 }
```

\@Glsuseri@ First letter uppercase version.

```
2754 \def\@Glsuseri@#1#2[#3]{%
2755   \glstrassignfieldfont{#2}%
2756   \@gls@field@link
2757   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2758   {\@gls@field@font{\Glsentryuseri{#2}#3}}%
2759 }
```

\@GLSuseri@ All uppercase version.

```
2760 \def\@GLSuseri@#1#2[#3]{%
2761   \glstrassignfieldfont{#2}%
2762   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2763   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseri{#2}#3}}}%
2764 }
```

\@Glsuserii@ First letter uppercase version.

```
2765 \def\@Glsuserii@#1#2[#3]{%
2766   \glstrassignfieldfont{#2}%
2767   \@gls@field@link
2768   [\let\glscapscase\@secondoftwo]%
2769   {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
2770 }
```

\@GLSuserii@ All uppercase version.

```
2771 \def\@GLSuserii@#1#2[#3]{%
2772   \glstrassignfieldfont{#2}%
2773   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2774   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
2775 }
```

\@Glsuseriii@ First letter uppercase version.

```
2776 \def\@Glsuseriii@#1#2[#3]{%
2777   \glstrassignfieldfont{#2}%
2778   \@gls@field@link
2779   [\let\glscapscase\@secondoftwo]%
2780   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
2781 }
```

\@GLSuseriii@ All uppercase version.

```
2782 \def\@GLSuseriii@#1#2[#3]{%
2783   \glstrassignfieldfont{#2}%
2784   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2785   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
2786 }
```

\@Glsuseriv@ First letter uppercase version.

```
2787 \def\@Glsuseriv@#1#2[#3]{%
2788   \glstrassignfieldfont{#2}%
2789 }
```



```

2789 \@gls@field@link
2790 [\let\glscapscase\@secondoftwo]%
2791 {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
2792 }

```

\@GLSuseriv@ All uppercase version.

```

2793 \def\@GLSuseriv@#1#2[#3]{%
2794 \glstrassignfieldfont{#2}%
2795 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2796 {#1}{#2}%
2797 {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
2798 }

```

\@Glsuserv@ First letter uppercase version.

```

2799 \def\@Glsuserv@#1#2[#3]{%
2800 \glstrassignfieldfont{#2}%
2801 \@gls@field@link
2802 [\let\glscapscase\@secondoftwo]%
2803 {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
2804 }

```

\@GLSuserv@ All uppercase version.

```

2805 \def\@GLSuserv@#1#2[#3]{%
2806 \glstrassignfieldfont{#2}%
2807 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2808 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
2809 }

```

\@Glsuservi@ First letter uppercase version.

```

2810 \def\@Glsuservi@#1#2[#3]{%
2811 \glstrassignfieldfont{#2}%
2812 \@gls@field@link
2813 [\let\glscapscase\@secondoftwo]%
2814 {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}%
2815 }

```

\@GLSuservi@ All uppercase version.

```

2816 \def\@GLSuservi@#1#2[#3]{%
2817 \glstrassignfieldfont{#2}%
2818 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2819 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
2820 }

```

Commands like \acrshort already set \glsifplural, but they don't set \glstrifwasfirstuse 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.

ase@acrshort@warn Warn user that they need to use to new abbreviation commands.

```

2821 \newcommand*{\@@glsxtr@base@acrcmd@warn}[2]{%
2822   \GlossariesExtraWarning{Base acronym command \string#1\space
2823     should not be used with new abbreviation definitions. Use
2824     \string#2\space instead}%
2825 }

```

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

```

2826 \let\@@glsxtr@base@acrcmd\@@glsxtr@base@acrcmd@warn

```

\@acrshort No case change.

```

2827 \def\@acrshort#1#2[#3]{%
2828   \@glsxtr@base@acrcmd\acrshort\glsxtrshort
2829   \glsdoifexists{#2}%
2830   {%
2831     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2832     \let\glsxtrifwasfirstuse\@secondoftwo
2833     \let\glsifplural\@secondoftwo
2834     \let\glscapscase\@firstofthree
2835     \let\glsinsert\@empty
2836     \def\glscustomtext{%
2837       \acronymfont{\glsaccessshort{#2}}#3%
2838     }%
2839     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2840   }%
2841   \glspostlinkhook
2842 }

```

\@Acrshort First letter uppercase.

```

2843 \def\@Acrshort#1#2[#3]{%
2844   \@glsxtr@base@acrcmd\Acrshort\Glsxtrshort
2845   \glsdoifexists{#2}%
2846   {%
2847     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2848     \let\glsxtrifwasfirstuse\@secondoftwo
2849     \let\glsifplural\@secondoftwo
2850     \let\glscapscase\@secondofthree
2851     \let\glsinsert\@empty
2852     \def\glscustomtext{%
2853       \acronymfont{\Glsaccessshort{#2}}#3%
2854     }%
2855     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2856   }%
2857   \glspostlinkhook
2858 }

```

\@ACRshort All uppercase.

```

2859 \def\@ACRshort#1#2[#3]{%
2860   \@glsxtr@base@acrcmd\ACRshort\GLSxtrshort
2861   \glsdoifexists{#2}%

```

```

2862 {%
2863   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2864   \let\glxtrifwasfirstuse\@secondoftwo
2865   \let\gl@sifplural\@secondoftwo
2866   \let\glscapscase\@thirdofthree
2867   \let\gl$insert\@empty
2868   \def\glscustomtext{%
2869     \mfirstucMakeUppercase{\acronymfont{\glaccessshort{#2}}#3}%
2870   }%
2871   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2872 }%
2873 \glspostlinkhook
2874 }

```

\@acrshortpl No case change.

```

2875 \def\@acrshortpl#1#2[#3]{%
2876   \@glxtr@base@acrcmd\acrshortpl\glxtrshortpl
2877   \glsoifexists{#2}%
2878   {%
2879     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2880     \let\glxtrifwasfirstuse\@secondoftwo
2881     \let\gl@sifplural\@firstoftwo
2882     \let\glscapscase\@firstofthree
2883     \let\gl$insert\@empty
2884     \def\glscustomtext{%
2885       \acronymfont{\glaccessshortpl{#2}}#3%
2886     }%
2887     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2888   }%
2889   \glspostlinkhook
2890 }

```

\@Acrshortpl First letter uppercase.

```

2891 \def\@Acrshortpl#1#2[#3]{%
2892   \@glxtr@base@acrcmd\Acrshortpl\Glsxtrshortpl
2893   \glsoifexists{#2}%
2894   {%
2895     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2896     \let\glxtrifwasfirstuse\@secondoftwo
2897     \let\gl@sifplural\@firstoftwo
2898     \let\glscapscase\@secondofthree
2899     \let\gl$insert\@empty
2900     \def\glscustomtext{%
2901       \acronymfont{\Glsaccessshortpl{#2}}#3%
2902     }%
2903     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2904   }%
2905   \glspostlinkhook
2906 }

```

\@ACRshortpl All uppercase.

```
2907 \def\@ACRshortpl#1#2[#3]{%
2908   \@glsxtr@base@acrcmd\ACRshortpl\GLSxtrshortpl
2909   \glsdoifexists{#2}%
2910   {%
2911     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2912     \let\glsxtrifwasfirstuse\@secondoftwo
2913     \let\glsifplural\@firstoftwo
2914     \let\glsupcase\@thirdofthree
2915     \let\glsinsert\@empty
2916     \def\glscustomtext{%
2917       \mfirstucMakeUppercase{\acronymfont{\glsaccessshortpl{#2}}#3}%
2918     }%
2919     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2920   }%
2921   \glspostlinkhook
2922 }
```

\@acrlong No case change.

```
2923 \def\@acrlong#1#2[#3]{%
2924   \@glsxtr@base@acrcmd\acrlong\glsxtrlong
2925   \glsdoifexists{#2}%
2926   {%
2927     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2928     \let\glsxtrifwasfirstuse\@secondoftwo
2929     \let\glsifplural\@secondoftwo
2930     \let\glsupcase\@firstofthree
2931     \let\glsinsert\@empty
2932     \def\glscustomtext{%
2933       \acronymfont{\glsaccesslong{#2}}#3%
2934     }%
2935     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2936   }%
2937   \glspostlinkhook
2938 }
```

\@Acrlong First letter uppercase.

```
2939 \def\@Acrlong#1#2[#3]{%
2940   \@glsxtr@base@acrcmd\Acrlong\Glsxtrlong
2941   \glsdoifexists{#2}%
2942   {%
2943     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2944     \let\glsxtrifwasfirstuse\@secondoftwo
2945     \let\glsifplural\@secondoftwo
2946     \let\glsupcase\@secondofthree
2947     \let\glsinsert\@empty
2948     \def\glscustomtext{%
2949       \acronymfont{\Glsaccesslong{#2}}#3%
2950     }%
2951   }
```

```

2951 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2952 }%
2953 \glspostlinkhook
2954 }

```

\@ACRlong All uppercase.

```

2955 \def\@ACRlong#1#2[#3]{%
2956 \glsxtr@base@acrcmd\ACRlong\GLSxtrlong
2957 \glsdoifexists{#2}%
2958 {%
2959 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2960 \let\glsxtrifwasfirstuse\@secondoftwo
2961 \let\glsifplural\@secondoftwo
2962 \let\glscapscase\@thirdofthree
2963 \let\glsinsert\@empty
2964 \def\glscustomtext{%
2965 \mfirstucMakeUppercase{\acronymfont{\glsaccesslong{#2}}#3}%
2966 }%
2967 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2968 }%
2969 \glspostlinkhook
2970 }

```

\@acrlongpl No case change.

```

2971 \def\@acrlongpl#1#2[#3]{%
2972 \glsxtr@base@acrcmd\acrlongpl\glsxtrlongpl
2973 \glsdoifexists{#2}%
2974 {%
2975 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2976 \let\glsxtrifwasfirstuse\@secondoftwo
2977 \let\glsifplural\@firstoftwo
2978 \let\glscapscase\@firstofthree
2979 \let\glsinsert\@empty
2980 \def\glscustomtext{%
2981 \acronymfont{\glsaccesslongpl{#2}}#3%
2982 }%
2983 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2984 }%
2985 \glspostlinkhook
2986 }

```

\@Acrlongpl First letter uppercase.

```

2987 \def\@Acrlongpl#1#2[#3]{%
2988 \glsxtr@base@acrcmd\Acrlongpl\Glsxtrlongpl
2989 \glsdoifexists{#2}%
2990 {%
2991 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2992 \let\glsxtrifwasfirstuse\@secondoftwo
2993 \let\glsifplural\@firstoftwo

```

```

2994 \let\glscapscase\@secondofthree
2995 \let\glsinsert\@empty
2996 \def\glscustomtext{%
2997     \acronymfont{\Glsaccesslongpl{#2}}#3%
2998 }%
2999 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3000 }%
3001 \glspostlinkhook
3002 }

```

\@ACRlongpl All uppercase.

```

3003 \def\@ACRlongpl#1#2[#3]{%
3004     \@glsxtr@base@acrcmd\ACRlongpl\GLSxtrlongpl
3005     \glsdoifexists{#2}%
3006     {%
3007         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3008         \let\glsxtrifwasfirstuse\@secondoftwo
3009         \let\glsifplural\@firstoftwo
3010         \let\glscapscase\@thirdofthree
3011         \let\glsinsert\@empty
3012         \def\glscustomtext{%
3013             \mfirstucMakeUppercase{\acronymfont{\Glsaccesslongpl{#2}}#3}%
3014         }%
3015         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3016     }%
3017     \glspostlinkhook
3018 }

```

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 \glsxtrfull etc) is present.

\@acrfull

```

3019 \def\@acrfull#1#2[#3]{%
3020     \@glsxtr@base@acrcmd\acrfull\glsxtrfull
3021     \acrfullfmt{#1}{#2}{#3}%
3022 }

```

\@Acrfull

```

3023 \def\@Acrfull#1#2[#3]{%
3024     \@glsxtr@base@acrcmd\Acrfull\Glsxtrfull
3025     \Acrfullfmt{#1}{#2}{#3}%
3026 }

```

\@ACRfull

```

3027 \def\@ACRfull#1#2[#3]{%

```

```

3028 \@glsxtr@base@acrcmd\ACRfull\GLSxtrfull
3029 \ACRfullfmt{#1}{#2}{#3}%
3030 }

```

\@acrfullpl

```

3031 \def\@acrfullpl#1#2[#3]{%
3032 \@glsxtr@base@acrcmd\acrfullpl\glsxtrfullpl
3033 \acrfullplfmt{#1}{#2}{#3}%
3034 }

```

\@Acrfullpl

```

3035 \def\@Acrfullpl#1#2[#3]{%
3036 \@glsxtr@base@acrcmd\Acrfullpl\Glsxtrfullpl
3037 \Acrfullplfmt{#1}{#2}{#3}%
3038 }

```

\@ACRfullpl

```

3039 \def\@ACRfullpl#1#2[#3]{%
3040 \@glsxtr@base@acrcmd\ACRfullpl\GLSxtrfullpl
3041 \ACRfullplfmt{#1}{#2}{#3}%
3042 }

```

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

\@glsaddkey

```

3043 \renewcommand*{\@glsaddkey}[7]{%
3044 \key@ifundefined{glossentry}{#1}%
3045 {%
3046 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
3047 \appto\@gls@keymap{,{#1}{#1}}%
3048 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
3049 \appto\@newglossaryentryposthook{%
3050 \letcs{@glo@tmp}{@glo@#1}%
3051 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
3052 }%
3053 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
3054 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

3055 \ifcsdef{@gls@user@#1@}%
3056 {%
3057 \PackageError{glossaries}%
3058 {Can't define '\string#5' as helper command
3059 '\expandafter\string\csname @gls@user@#1@\endcsname' already
3060 exists}%
3061 }%
3062 }%
3063 {%

```

```

3064 \expandafter\newcommand\expandafter*\expandafter
3065   {\csname @gls@user@#1\endcsname}[2][\%
3066     \new@ifnextchar[%
3067       {\csuse{@gls@user@#1@}{##1}{##2}}%
3068       {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
3069   \csdef{@gls@user@#1@}##1##2[##3]{%
3070     \@gls@field@link{##1}{##2}{#3{##2}##3}%
3071   }%
3072   \newrobustcmd*{#5}{%
3073     \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
3074   }%

```

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

```

3075 \ifcsdef{@Gls@user@#1@}%
3076 {%
3077   \PackageError{glossaries}%
3078   {Can't define '\string#6' as helper command
3079     '\expandafter\string\csname @Gls@user@#1\endcsname' already
3080     exists}%
3081   }%
3082 }%
3083 {%
3084   \expandafter\newcommand\expandafter*\expandafter
3085     {\csname @Gls@user@#1\endcsname}[2][\%
3086       \new@ifnextchar[%
3087         {\csuse{@Gls@user@#1@}{##1}{##2}}%
3088         {\csuse{@Gls@user@#1@}{##1}{##2}[]}}%
3089     \csdef{@Gls@user@#1@}##1##2[##3]{%
3090       \@gls@field@link[\let\glscaps@case\@secondofthree]%
3091       {##1}{##2}{#4{##2}##3}%
3092     }%
3093     \newrobustcmd*{#6}{%
3094       \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
3095     }%

```

Finally the all caps version (modified):

```

3096 \ifcsdef{@GLS@user@#1@}%
3097 {%
3098   \PackageError{glossaries}%
3099   {Can't define '\string#7' as helper command
3100     '\expandafter\string\csname @GLS@user@#1\endcsname' already
3101     exists}%
3102   }%
3103 }%
3104 {%
3105   \expandafter\newcommand\expandafter*\expandafter
3106     {\csname @GLS@user@#1\endcsname}[2][\%
3107       \new@ifnextchar[%
3108         {\csuse{@GLS@user@#1@}{##1}{##2}}%
3109         {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%

```



```

3110 \csdef{@GLS@user@#1@}##1##2[##3]{%
3111 \@gls@field@link[\let\glscapscase\@thirdofthree]%
3112 {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
3113 }%
3114 \newrobustcmd*{#7}{%
3115 \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
3116 }%
3117 }%
3118 {%
3119 \PackageError{glossaries-extra}{Key ‘#1’ already exists}{}%
3120 }%
3121 }

```

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

```

3122 \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.

```

3123 \let\@glxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
3124 \renewcommand*{\@gls@link@checkfirsthyper}{%

```

\ifglused 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 \glxtrifwasfirstuse to \@secondoftwo (which is done in \@glxtr@field@linkdefs). Note that if the entry is undefined (as with bib2gls on the first \LaTeX run), \ifglused 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).

```

3125 \ifglused{\glslabel}%
3126 {\let\glxtrifwasfirstuse\@secondoftwo}
3127 {\let\glxtrifwasfirstuse\@firstoftwo}%

```

Store the category label for convenience.

```

3128 \protected@edef\glscategorylabel{\glscategory{\glslabel}}%
3129 \ifglused{\glslabel}%
3130 {%
3131 \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
3132 {\KV@glslink@hyperfalse}}%
3133 }%
3134 {%
3135 \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
3136 {\KV@glslink@hyperfalse}}%
3137 }%
3138 \glslinkcheckfirsthyperhook
3139 }

```

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.

```

3140 \ifdef\do@gl:disablehyperinlist
3141 {%
3142   \let\@gl:extr@do@gl:disablehyperinlist\do@gl:disablehyperinlist
3143   \renewcommand*{\do@gl:disablehyperinlist}{%
3144     \@gl:extr@do@gl:disablehyperinlist
3145     \gl:ifattribute{\gl:label}{nohyper}{true}{\KV@gl:link@hyperfalse}}}%
3146   }
3147 }
3148 {}

```

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

```

3149 \define@boolkey{gl:link}{noindex}[true]{}
3150 \KV@gl:link@noindexfalse

```

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

lt@gl:link@opts

```

3151 \ifdef\@gl:setdefault@gl:link@opts
3152 {
3153   \renewcommand*{\@gl:setdefault@gl:link@opts}{%
3154     \KV@gl:link@noindexfalse
3155     \@gl:extr:setaliasnoindex
3156   }
3157 }
3158 {

```

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

```

3159   \newcommand*{\@gl:setdefault@gl:link@opts}{%
3160     \KV@gl:link@noindexfalse
3161     \@gl:extr:setaliasnoindex
3162   }
3163   \pretol\do@gl:disablehyperinlist{\@gl:setdefault@gl:link@opts}
3164 }

```

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.)

```

3165 \providecommand*{\gl:extr:setaliasnoindex}{%
3166   \KV@gl:link@noindextrue
3167 }

```

setaliasnoindex

```

3168 \newcommand*{\@gl:extr:setaliasnoindex}{%
3169   \s@gl:extr:ifhasfield{alias}{\gl:label}%
3170   {%
3171     \let\gl:extr:indexaliased\gl:extr:indexaliased
3172     \gl:extr:setaliasnoindex
3173     \let\gl:extr:indexaliased\@no@gl:extr:indexaliased
3174   }%

```

```

3175 {}%
3176 }

```

xtrindexaliased

```

3177 \newcommand{\@glxtrindexaliased}{%
3178   \ifKV@glslink@noindex
3179   \else
3180     \begingroup
3181     \let\@glxnumberformat\@glxtr@defaultnumberformat

3182     \protected@edef\@glx@counter{\csname glo@\glx@detoklabel{\glxlabel}\@counter\endcsname}%
3183     \glxtr@saveentrycounter
3184     \@do@wrglossary{\glxtralias{\glxlabel}}%
3185     \endgroup
3186   \fi
3187 }

```

xtrindexaliased

```

3188 \newcommand{\@no@glxtrindexaliased}{%
3189   \PackageError{glossaries-extra}{\string\glxtrindexaliased\space
3190     not permitted outside definition of \string\glxtrsetaliasnoindex}%
3191   {}%
3192 }

```

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

```

3193 \let\glxtrindexaliased\@no@glxtrindexaliased

```

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

```

3194 \newcommand*\@GlsXtrSetDefaultGlsOpts}[1]{%
3195   \renewcommand*\@glx@setdefault@glslink@opts}{%
3196     \setkeys{glslink}{#1}%
3197     \@glxtrsetaliasnoindex
3198   }%
3199 }

```

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

```

3200 \newcommand*\@glxtrifindexing}[2]{%
3201   \ifKV@glslink@noindex #2\else #1\fi
3202 }

```

\glswriteentry Redefine to test for `indexonlyfirst` category attribute. This needs to use `\GlsXtrIfUnusedOrUndefined` instead of `\ifglxused` to allow it to work with `bib2gls`.

```

3203 \renewcommand*\@glswriteentry}[2]{%
3204   \@glxtrifindexing
3205   {%
3206     \ifglxindexonlyfirst
3207       \GlsXtrIfUnusedOrUndefined{#1}
3208       {#2}%

```

```

3209     {\glxtrdoautoindexname{#1}{dualindex}}}%
3210 \else
3211     \glxifattribute{#1}{indexonlyfirst}{true}%
3212     {%
3213         \GlsXtrIfUnusedOrUndefined{#1}%
3214         {#2}%
3215         {\glxtrdoautoindexname{#1}{dualindex}}}%
3216     }%
3217     {#2}%
3218 \fi
3219 }%
3220 {}%
3221 }

```

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

```

3222 \appto\@do@wrglossary{\@glxtr@do@wrindex
3223   \glxtrdowrglossaryhook{\@gls@label}%
3224 }

```

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

Similarly for the “noidx” version:

`s@noidxglossary`

```

3225 \appto\gls@noidxglossary{\@glxtr@do@wrindex
3226   \glxtrdowrglossaryhook{\@gls@label}%
3227 }

```

`xtr@do@wrindex`

```

3228 \newcommand*{\@glxtr@do@wrindex}{%
3229   \glxtrdoautoindexname{\@gls@label}{dualindex}%
3230 }

```

`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.)

```

3231 \newcommand*{\glxtrdowrglossaryhook}[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.

```

3232 \newcommand*{\@gls@alt@hyp@opt}[1]{%
3233   \let\glslinkvar\@firstofthree
3234   \let\@gls@hyp@opt@cs#1\relax
3235   \@ifstar{\s@gls@hyp@opt}%
3236   {\@ifnextchar+%
3237     {\@firstoftwo{\p@gls@hyp@opt}}%
3238     {%
3239       \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
3240       {\@firstoftwo{\@alt@gls@hyp@opt}}%

```

```

3241      {#1}%
3242    }%
3243  }%
3244 }

```

`\alt@glshyp@opt` User version

```

3245 \newcommand*\@alt@glshyp@opt}[1][]{%
3246   \let\glslinkvar\@firstofthree
3247   \expandafter\@glshyp@opt@cs\expandafter[\@glshyp@opt@keys,#1]}

```

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

```

3248 \newcommand*\@glshyp@opt@char{}

```

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

```

3249 \newcommand*\@glshyp@opt@keys{}

```

`\rSetAltModifier`

```

3250 \newcommand*\GlsXtrSetAltModifier}[2]{%
3251   \let\@glshyp@opt\@glshyp@opt
      Check that the supplied character isn't + or *
3252   \ifstrequal{#1}{+}%
3253   {\PackageError{glossaries-extra}%
3254     {Can't use '#1' as modifier (it's already in use)}}}%
3255   {%
3256     \ifstrequal{#1}{*}%
3257     {\PackageError{glossaries-extra}%
3258       {Can't use '#1' as modifier (it's already in use)}}}%
3259   }%
3260 }%
3261 \def\@glshyp@opt@char{#1}%
3262 \def\@glshyp@opt@keys{#2}%
3263 \ifdefequal\@glsxtr@record@setting\@glsxtr@record@setting@off
3264 {}%
3265 {%
      Let bib2gls know the modifier.
3266   \protected@write\@auxout{}\string\providecommand{\string\@glsxtr@altmodifier}[1]{}%
3267   \protected@write\@auxout{}\string\@glsxtr@altmodifier{#1}%
3268 }%
3269 }

```

`\org@dohyperlink`

```

3270 \let\glsxtr@org@dohyperlink\glsdohyperlink

```

`\glshyp@opt@char` Since `\glshyp@opt@char` 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.

```

3271 \ifdef\glsnavhyperlink
3272 {
3273   \renewcommand*{\glsnavhyperlink}[3][\@glo@type]{%
3274     \protected@edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%

```

Scope:

```

3275   {%
3276     \let\glsxtrdohyperlink\glsxtr@org@dohyperlink
3277     \@glslink{\glsnavhyperlinkname{#1}{#2}}{#3}%
3278   }%
3279 }%
3280 }
3281 {}

```

The redefinition of `\glsdohyperlink` has been causing problems so introduce a new command instead.

`\glsxtrdohyperlink` 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 `\glsxtrshort` to use `\glsentryshort` and, similarly, the long form commands like `\acrlong` and `\glsxtrlong` to use `\glsentrylong`. Added attribute check.

```

3282 \newcommand*{\glsxtrdohyperlink}[2]{%
3283   \glsattribute{\glslabel}{targeturl}%
3284   {%
3285     \glsattribute{\glslabel}{targetname}%
3286     {%
3287       \glsattribute{\glslabel}{targetcategory}%
3288       {%
3289         \hyperref{\glsattribute{\glslabel}{targeturl}}{
3290           \glsattribute{\glslabel}{targetcategory}}%
3291         \glsattribute{\glslabel}{targetname}}%
3292         {\glsxtrprotectlinks#2}}%
3293       }%
3294     }%
3295     \hyperref{\glsattribute{\glslabel}{targeturl}}{
3296       }%
3297       \glsattribute{\glslabel}{targetname}}%
3298       {\glsxtrprotectlinks#2}}%
3299   }%
3300 }%
3301 {%
3302   \href{\glsattribute{\glslabel}{targeturl}}{
3303     {\glsxtrprotectlinks#2}}%
3304   }%
3305 }%

```

```

3306 {%
    Check for alias.
3307     \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
3308     \ifdefvoid\gloaliaslabel
3309     {%
3310         \glsxtrhyperlink{#1}{\glsxtrprotectlinks#2}}%
3311     }%
3312 {%

```

Redirect link to the alias target.

```

3313     \glsxtrhyperlink
3314     {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
3315     {\glsxtrprotectlinks#2}}%
3316 }%
3317 }%
3318 }

```

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

```

3319 \ifdef\@glsshowtarget
3320 {
3321     \newcommand{\glsxtrhyperlink}[2]{%
3322         \@glsshowtarget{#1}%
3323         \hyperlink{#1}{#2}%
3324     }%
3325 }
3326 {
3327     \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%
3328 }

```

`\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.

```

3329 \renewrobustcmd*{\gls hyperlink}[2][\glsentrytext{\@glo@label}]{%
3330     \glsdoifexists{#2}%
3331     {%
3332         \def\@glo@label{#2}%
3333         {\protected@edef\glslabel{#2}%
3334         \@glslink{\glolinkprefix\glslabel}{#1}}%
3335     }%
3336 }

```

`\glsdisablehyper` Redefine in case we have an old version of glossaries. This now uses `\def` rather than `\let` to allow for redefinitions of `\glsdohyperlink`.

```

3337 \renewcommand{\glsdisablehyper}{%
3338     \KV@glslink@hyperfalse
3339     \def\@glslink{\glsdohyperlink}%

```

```

3340 \let\@glstarget\@secondoftwo
3341 }

```

`\glsenablehyper` This now uses `\def` rather than `\let` to allow for redefinitions of `\glsdohypertarget` and `\glsdohyperlink`.

```

3342 \renewcommand{\glsenablehyper}{%
3343 \KV@glslink@hypertrue
3344 \def\@glslink{\glsxtrdohyperlink}%
3345 \def\@glstarget{\glsdohypertarget}%
3346 }

```

`\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).

```

3347 \def\glsdonohyperlink#1#2{{\glsxtrprotectlinks #2}}

```

`\@glslink` Reset `\@glslink` with patched versions:

```

3348 \ifcsundef{hyperlink}%
3349 {%
3350 \def\@glslink{\glsdonohyperlink}
3351 }%
3352 {%
3353 \def\@glslink{\glsxtrdohyperlink}
3354 }

```

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

```

3355 \newcommand*{\glsxtrprotectlinks}{%
3356 \KV@glslink@hyperfalse
3357 \KV@glslink@noindextrue
3358 \let\@gls@\@glsxtr@p@text@
3359 \let\@Gls@\@Glsxtr@p@text@
3360 \let\@GLS@\@GLSxtr@p@text@
3361 \let\@glspl@\@glsxtr@p@plural@
3362 \let\@Glspl@\@Glsxtr@p@plural@
3363 \let\@GLSpl@\@GLSxtr@p@plural@
3364 \let\@glsxtrshort@\@glsxtr@p@short@
3365 \let\@Glsxtrshort@\@Glsxtr@p@short@
3366 \let\@GLSxtrshort@\@GLSxtr@p@short@
3367 \let\@glsxtrlong@\@glsxtr@p@long@
3368 \let\@Glsxtrlong@\@Glsxtr@p@long@
3369 \let\@GLSxtrlong@\@GLSxtr@p@long@
3370 \let\@glsxtrshortpl@\@glsxtr@p@shortpl@
3371 \let\@Glsxtrshortpl@\@Glsxtr@p@shortpl@
3372 \let\@GLSxtrshortpl@\@GLSxtr@p@shortpl@
3373 \let\@glsxtrlongpl@\@glsxtr@p@longpl@
3374 \let\@Glsxtrlongpl@\@Glsxtr@p@longpl@

```



```

3375 \let\@GLSxtrlongpl\@GLSxtrp@longpl@
3376 \let\@acrshort\@glxstrp@acrshort@
3377 \let\@Acrshort\@Glsxtrp@acrshort@
3378 \let\@ACRshort\@GLSxtrp@acrshort@
3379 \let\@acrshortpl\@glxstrp@acrshortpl@
3380 \let\@Acrshortpl\@Glsxtrp@acrshortpl@
3381 \let\@ACRshortpl\@GLSxtrp@acrshortpl@
3382 \let\@acrlong\@glxstrp@acrlong@
3383 \let\@Acrlong\@Glsxtrp@acrlong@
3384 \let\@ACRlong\@GLSxtrp@acrlong@
3385 \let\@acrlongpl\@glxstrp@acrlongpl@
3386 \let\@Acrlongpl\@Glsxtrp@acrlongpl@
3387 \let\@ACRlongpl\@GLSxtrp@acrlongpl@
3388 }

```

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

@glxstrp@text@

```

3389 \def\@glxstrp@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}

```

@Glsxtrp@text@

```

3390 \def\@Glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}

```

@GLSxtrp@text@

```

3391 \def\@GLSxtrp@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}

```

lsxtrp@plural@

```

3392 \def\@glxstrp@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}

```

lsxtrp@plural@

```

3393 \def\@Glsxtrp@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}

```

LSxtrp@plural@

```

3394 \def\@GLSxtrp@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}

```

glxstrp@short@

```

3395 \def\@glxstrp@short@#1#2[#3]{%
3396 {%
3397   \glssetabbrvfmt{\glscategory{#2}}%
3398   \glsabbrvfont{\glsentryshort{#2}}#3%
3399 }%
3400 }

```

Glsxtrp@short@

```

3401 \def\@Glsxtrp@short@#1#2[#3]{%
3402 {%
3403   \glssetabbrvfmt{\glscategory{#2}}%
3404   \glsabbrvfont{\Glsentryshort{#2}}#3%
3405 }%
3406 }

```

GLSxtr@p@short@

```
3407 \def\@GLSxtr@p@short@#1#2[#3]{%
3408   {%
3409     \glsetabbrvfmt{\glscategory{#2}}%
3410     \mfirstucMakeUppercase{\glabbrvfont{\glentryshort{#2}}#3}%
3411   }%
3412 }
```

sxtr@p@shortpl@

```
3413 \def\@glxtr@p@shortpl@#1#2[#3]{%
3414   {%
3415     \glsetabbrvfmt{\glscategory{#2}}%
3416     \glabbrvfont{\glentryshortpl{#2}}#3%
3417   }%
3418 }
```

Sxtr@p@shortpl@

```
3419 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
3420   {%
3421     \glsetabbrvfmt{\glscategory{#2}}%
3422     \glabbrvfont{\Glsentryshortpl{#2}}#3%
3423   }%
3424 }
```

Sxtr@p@shortpl@

```
3425 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
3426   {%
3427     \glsetabbrvfmt{\glscategory{#2}}%
3428     \mfirstucMakeUppercase{\glabbrvfont{\glentryshortpl{#2}}#3}%
3429   }%
3430 }
```

@glxtr@p@long@

```
3431 \def\@glxtr@p@long@#1#2[#3]{\{\glentrylong{#2}#3}}
```

@Glsxtr@p@long@

```
3432 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@GLSxtr@p@long@

```
3433 \def\@GLSxtr@p@long@#1#2[#3]{%
3434   {\mfirstucMakeUppercase{\glslongfont{\glentrylong{#2}}#3}}}
```

lsxtr@p@longpl@

```
3435 \def\@glxtr@p@longpl@#1#2[#3]{\{\glentrylongpl{#2}#3}}
```

lsxtr@p@longpl@

```
3436 \def\@Glsxtr@p@longpl@#1#2[#3]{\{\glslongfont{\Glsentrylongpl{#2}}#3}}
```

```

LSxtr@p@longpl@
3437 \def\@GLSxtr@p@longpl@#1#2[#3]{%
3438   {\mfirstucMakeUppercase{\glslongfont{\glsentrylongpl{#2}}#3}}}

xtr@p@acrshort@
3439 \def\@glsxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\glsentryshort{#2}}#3}}

xtr@p@acrshort@
3440 \def\@GLSxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
3441 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
3442   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
3443 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
3444 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
3445 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
3446   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}

sxtr@p@acrlong@
3447 \def\@glsxtr@p@acrlong@#1#2[#3]{\{\glsentrylong{#2}}#3}}

sxtr@p@acrlong@
3448 \def\@GLSxtr@p@acrlong@#1#2[#3]{\{\Glsentrylong{#2}}#3}}

Sxtr@p@acrlong@
3449 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
3450   {\mfirstucMakeUppercase{\glsentrylong{#2}}#3}}}

tr@p@acrlongpl@
3451 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\{\glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
3452 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{\{\Glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
3453 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
3454   {\mfirstucMakeUppercase{\glsentrylongpl{#2}}#3}}}

Commands to minimise conflict.

\@glsxtrp@opt
3455 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

```

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

```
3456 \newcommand*{\glsxtrsetpopts}[1]{%
3457   \renewcommand*{\@glsxtrp@opt}{#1}%
3458 }
```

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

```
3459 \newcommand*{\glossxtrsetpopts}{%
3460   \glsxtrsetpopts{noindex}%
3461 }
```

`\@@glsxtrp`

```
3462 \newrobustcmd*{\@@glsxtrp}[2]{%
    Add scope.
3463   {%
3464     \let\glspostlinkhook\relax
3465     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
3466   }%
3467 }
```

`\@glsxtrp`

```
3468 \newrobustcmd*{\@glsxtrp}[2]{%
3469   \ifcsdef{gls#1}%
3470     {%
3471       \@glsxtrp{gls#1}{#2}%
3472     }%
3473     {%
3474       \ifcsdef{glsxtr#1}%
3475         {%
3476           \@glsxtrp{glsxtr#1}{#2}%
3477         }%
3478         {%
3479           \PackageError{glossaries-extra}{‘#1’ not recognised by
3480             \string\glsxtrp}{}%
3481         }%
3482       }%
3483 }
```

`\@Glsxtrp`

```
3484 \newrobustcmd*{\@Glsxtrp}[2]{%
3485   \ifcsdef{Gls#1}%
3486     {%
3487       \@glsxtrp{Gls#1}{#2}%
3488     }%
3489     {%
3490       \ifcsdef{Glsxtr#1}%
3491         {%
3492           \@glsxtrp{Glsxtr#1}{#2}%
3493         }%
3494     }
```

```

3494   {%
3495       \PackageError{glossaries-extra}{‘#1’ not recognised by
3496       \string\Glsxtrp}{}%
3497   }%
3498 }%
3499 }

```

\@GLSxtrp

```

3500 \newrobustcmd*{\@GLSxtrp}[2]{%
3501   \ifcsdef{GLS#1}%
3502   {%
3503       \@@glxtrp{GLS#1}{#2}%
3504   }%
3505   {%
3506       \ifcsdef{GLSxtr#1}%
3507       {%
3508           \@@glxtrp{GLSxtr#1}{#2}%
3509       }%
3510       {%
3511           \PackageError{glossaries-extra}{‘#1’ not recognised by
3512           \string\GLSxtrp}{}%
3513       }%
3514   }%
3515 }

```

\glxtr@entry@p

```

3516 \newrobustcmd*{\glxtr@headentry@p}[2]{%
3517   \gl@ifattribute{#1}{headuc}{true}%
3518   {%
3519       \mfirstucMakeUppercase{\@glx@entry@field{#1}{#2}}%
3520   }%
3521   {%
3522       \@glx@entry@field{#1}{#2}%
3523   }%
3524 }

```

\glxtrp Not robust as it needs to expand somewhat.

```

3525 \ifdef\texorpdfstring
3526 {
3527   \newcommand{\glxtrp}[2]{%
3528     \protect\NoCaseChange
3529     {%
3530         \protect\texorpdfstring
3531         {%
3532             \protect\glxtrifinmark
3533             {%
3534                 \ifcsdef{glxtrhead#1}%
3535                 {%
3536                     {\protect\csuse{glxtrhead#1}{#2}}%

```

```

3537         }%
3538         {%
3539         \glxtr@headentry@p{#2}{#1}%
3540         }%
3541     }%
3542     {%
3543     \@glxtrp{#1}{#2}%
3544     }%
3545 }%
3546 {%
3547 \protect\@gls@entry@field{#2}{#1}%
3548 }%
3549 }%
3550 }
3551 }
3552 {
3553 \newcommand{\glxtrp}[2]{%
3554 \protect\NoCaseChange
3555 {%
3556 \protect\glxtrifinmark
3557 {%
3558 \ifcsdef{glxtrhead#1}%
3559 {%
3560 {\protect\csuse{glxtrhead#1}}%
3561 }%
3562 {%
3563 \glxtr@headentry@p{#2}{#1}%
3564 }%
3565 }%
3566 {%
3567 \@glxtrp{#1}{#2}%
3568 }%
3569 }%
3570 }
3571 }

```

Provide short synonyms for the most common option.

`\glsp`

```

3572 \newcommand*{\glsp}{\glxtrp{short}}

```

`\glsp`

```

3573 \newcommand*{\glsp}{\glxtrp{text}}

```

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

```

3574 \ifdef\texorpdfstring
3575 {
3576 \newcommand{\Glsxtrp}[2]{%

```

```

3577 \protect\NoCaseChange
3578 {%
3579 \protect\texorpdfstring
3580 {%
3581 \protect\glxtrifinmark
3582 {%
3583 \ifcsdef{Glsxtrhead#1}%
3584 {%
3585 {\protect\csuse{Glsxtrhead#1}{#2}}%
3586 }%
3587 {%
3588 \protect\@Gls@entry@field{#2}{#1}%
3589 }%
3590 }%
3591 {%
3592 \@Glsxtrp{#1}{#2}%
3593 }%
3594 }%
3595 {%
3596 \protect\@gls@entry@field{#2}{#1}%
3597 }%
3598 }%
3599 }
3600 }
3601 {
3602 \newcommand{\Glsxtrp}[2]{%
3603 \protect\NoCaseChange
3604 {%
3605 \protect\glxtrifinmark
3606 {%
3607 \ifcsdef{Glsxtrhead#1}%
3608 {%
3609 {\protect\csuse{Glsxtrhead#1}}%
3610 }%
3611 {%
3612 \protect\@Gls@entry@field{#2}{#1}%
3613 }%
3614 }%
3615 {%
3616 \@Glsxtrp{#1}{#2}%
3617 }%
3618 }%
3619 }
3620 }

```

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

```

3621 \ifdef\texorpdfstring
3622 {
3623 \newcommand{\Glsxtrp}[2]{%

```

```

3624 \protect\NoCaseChange
3625 {%
3626 \protect\texorpdfstring
3627 {%
3628 \protect\glstrifinmark
3629 {%
3630 \ifcsdef{GLSxtr#1}%
3631 {%
3632 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
3633 }%
3634 {%
3635 \protect\mfirstucMakeUppercase
3636 {%
3637 \protect\@gls@entry@field{#2}{#1}%
3638 }%
3639 }%
3640 }%
3641 {%
3642 \@GLSxtrp{#1}{#2}%
3643 }%
3644 }%
3645 {%
3646 \protect\@gls@entry@field{#2}{#1}%
3647 }%
3648 }%
3649 }
3650 }
3651 {
3652 \newcommand{\GLSxtrp}[2]{%
3653 \protect\NoCaseChange
3654 {%
3655 \protect\glstrifinmark
3656 {%
3657 \ifcsdef{GLSxtr#1}%
3658 {%
3659 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
3660 }%
3661 {%
3662 \protect\mfirstucMakeUppercase
3663 {%
3664 \protect\@gls@entry@field{#2}{#1}%
3665 }%
3666 }%
3667 }%
3668 {%
3669 \@GLSxtrp{#1}{#2}%
3670 }%
3671 }%
3672 }

```


3673 }

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's 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 be switched off by redefining \@glsunset and then switched back on again by changing the definition back to \@glsxtr@unset.

\@glsxtr@unset Global unset.

```
3674 \newcommand*{\@glsxtr@unset}[1]{%
3675   \@glsunset{#1}%
3676   \glsxtrpostunset{#1}%
3677 }
```

\@glsunset Global unset.

```
3678 \let\@glsunset\@glsxtr@unset
```

glsxtrpostunset

```
3679 \newcommand*{\glsxtrpostunset}[1]{}
```

Provide a command to store a list of labels that will need unsetting.

tUnsetBuffering

```
3680 \newcommand*{\GlsXtrStartUnsetBuffering}{%
3681   \ifstar\s@GlsXtrStartUnsetBuffering\@GlsXtrStartUnsetBuffering
3682 }
```

tUnsetBuffering Unstarred version doesn't check for duplicates.

```
3683 \newcommand*{\@GlsXtrStartUnsetBuffering}{%
3684   \let\@glsxtr@org@unset@buffer\@glsxtr@unset@buffer
3685   \def\@glsxtr@unset@buffer{}%
3686   \let\@glsunset\@glsxtrbuffer@unset
3687 }
```

tUnsetBuffering Starred version checks for duplicates.

```
3688 \newcommand*{\s@GlsXtrStartUnsetBuffering}{%
3689   \let\@glsxtr@org@unset@buffer\@glsxtr@unset@buffer
3690   \def\@glsxtr@unset@buffer{}%

```

```

3691 \let\@glsunset\@glsxtrbuffer@nodup@unset
3692 }

xtrbuffer@unset This must use a global change since \gls may have to be placed inside \mbox (for example,
with soul commands).

3693 \newcommand*{\@glsxtrbuffer@unset}[1]{%
3694 \listxadd\@glsxtr@unset@buffer{#1}%
3695 }

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.)

3696 \newcommand*{\@glsxtrbuffer@nodup@unset}[1]{%
3697 \expandafter\ifinlist\expandafter{#1}{\@glsxtr@unset@buffer}{}%
3698 {\listxadd\@glsxtr@unset@buffer{#1}}%
3699 }

pUnsetBuffering

3700 \newcommand*{\GlsXtrStopUnsetBuffering}{%
3701 \@ifstar\s@GlsXtrStopUnsetBuffering\@GlsXtrStopUnsetBuffering
3702 }

pUnsetBuffering Unstarred form (global unset).

3703 \newcommand*{\@GlsXtrStopUnsetBuffering}{%
3704 \let\@glsunset\@glsxtr@unset
3705 \forlistloop\@glsunset\@glsxtr@unset@buffer
3706 \let\@glsxtr@unset@buffer\@glsxtr@org@unset@buffer
3707 }

pUnsetBuffering Starred form (local unset).

3708 \newcommand*{\s@GlsXtrStopUnsetBuffering}{%
3709 \forlistloop\@glslocalunset\@glsxtr@unset@buffer
3710 \let\@glsunset\@glsxtr@unset
3711 }

dUnsetBuffering Discards pending buffer and restores \glsunset.

3712 \newcommand*{\GlsXtrDiscardUnsetBuffering}{%
3713 \let\@glsunset\@glsxtr@unset
3714 \let\@glsxtr@unset@buffer\@glsxtr@org@unset@buffer
3715 }

setBufferedList Iterate over labels stored in the current buffer. The argument is the handler macro.

3716 \newcommand*{\GlsXtrForUnsetBufferedList}[1]{%
3717 \forlistloop#1\@glsxtr@unset@buffer
3718 }

```

```

\@glslocalunset   Local unset.
3719 \renewcommand*{\@glslocalunset}[1]{%
3720   \@glslocalunset{#1}%
3721   \glsxtrpostlocalunset{#1}%
3722 }%

rpostlocalunset
3723 \newcommand*{\glsxtrpostlocalunset}[1]{%

\@glsreset   Global reset.
3724 \renewcommand*{\@glsreset}[1]{%
3725   \@glsreset{#1}%
3726   \glsxtrpostreset{#1}%
3727 }%

glsxtrpostreset
3728 \newcommand*{\glsxtrpostreset}[1]{%

\@glslocalreset   Local reset.
3729 \renewcommand*{\@glslocalreset}[1]{%
3730   \@glslocalreset{#1}%
3731   \glsxtrpostlocalreset{#1}%
3732 }%

rpostlocalreset
3733 \newcommand*{\glsxtrpostlocalreset}[1]{%

slocalreseteach   Locally reset a list of entries.
3734 \newcommand*{\glslocalreseteach}[1]{%
3735   \gls@ifnotmeasuring
3736   {%
3737     \@for\@gls@thislabel:=#1\do{%
3738       \glsdoifexists{\@gls@thislabel}%
3739       {%
3740         \@glslocalreset{\@gls@thislabel}%
3741       }%
3742     }%
3743   }%
3744 }

slocalunseteach   Locally unset a list of entries.
3745 \newcommand*{\glslocalunseteach}[1]{%
3746   \gls@ifnotmeasuring
3747   {%
3748     \@for\@gls@thislabel:=#1\do{%
3749       \glsdoifexists{\@gls@thislabel}%
3750       {%
3751         \@glslocalunset{\@gls@thislabel}%

```

```

3752     }%
3753     }%
3754 }%
3755 }

```

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

```

3756 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%

```

Enable entry counting:

```

3757   \glsenableentrycount

```

Redefine `\gls` etc:

```

3758   \renewcommand*{\gls}{\cglsl}%
3759   \renewcommand*{\Gls}{\cGls}%
3760   \renewcommand*{\glspl}{\cglspl}%
3761   \renewcommand*{\Glspl}{\cGlspl}%
3762   \renewcommand*{\GLS}{\cGLS}%
3763   \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

3764   \@glsxtr@setentrycountunsetattr{#1}{#2}%

```

In case this command is used again:

```

3765   \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
3766   \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
3767     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
3768       can't be used with \string\GlsXtrEnableEntryCounting}%
3769     {Use one or other but not both commands}}%
3770 }

```

`ycountunsetattr`

```

3771 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
3772   \@for\@glsxtr@cat:=#1\do
3773   {%
3774     \ifdefempty{\@glsxtr@cat}{}%
3775     {%
3776       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3777     }%
3778   }%
3779 }

```

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

`nableentrycount`

```

3780 \renewcommand*{\glsenableentrycount}{%

```

Enable new fields:

```

3781   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%

```

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

```

3782 \renewcommand*{\gls@defdocnewglossaryentry}{%
3783 \renewcommand*\newglossaryentry[2]{%
3784 \PackageError{glossaries}{\string\newglossaryentry\space
3785 may only be used in the preamble when entry counting has
3786 been activated}{If you use \string\glsenableentrycount\space
3787 you must place all entry definitions in the preamble not in
3788 the document environment}%
3789 }%
3790 }%

```

New commands to access new fields:

```

3791 \newcommand*{\glsentrycurrcount}[1]{%
3792 \ifcsundef{glo@\glsdetoklabel{##1}@currcount}%
3793 {0}{\@gls@entry@field{##1}{currcount}}}%
3794 }%
3795 \newcommand*{\glsentryprevcount}[1]{%
3796 \ifcsundef{glo@\glsdetoklabel{##1}@prevcount}%
3797 {0}{\@gls@entry@field{##1}{prevcount}}}%
3798 }%

```

Adjust post unset and reset:

```

3799 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
3800 \renewcommand*{\glsxtrpostunset}[1]{%
3801 \@glsxtr@entrycount@org@unset{##1}%
3802 \@gls@increment@currcount{##1}%
3803 }%
3804 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
3805 \renewcommand*{\glsxtrpostlocalunset}[1]{%
3806 \@glsxtr@entrycount@org@localunset{##1}%
3807 \@gls@local@increment@currcount{##1}%
3808 }%
3809 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
3810 \renewcommand*{\glsxtrpostreset}[1]{%
3811 \@glsxtr@entrycount@org@reset{##1}%
3812 \csgdef{glo@\glsdetoklabel{##1}@currcount}{0}%
3813 }%
3814 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
3815 \renewcommand*{\glsxtrpostlocalreset}[1]{%
3816 \@glsxtr@entrycount@org@localreset{##1}%
3817 \csdef{glo@\glsdetoklabel{##1}@currcount}{0}%
3818 }%

```

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

```

3819 \let\@cgl@s@\@cgl@s@
3820 \let\@cgl spl@\@cgl spl@

3821 \let\@cGls@\@cGls@
3822 \let\@cGlspl@\@cGlspl@
3823 \let\@cGLS@\@cGLS@

```

```
3824 \let\@cGLSpl@ \@cGLSpl@
```

The rest is as the original definition.

```
3825 \AtEndDocument{\@gls@write@entrycounts}%
3826 \renewcommand*{\@gls@entry@count}[2]{%
3827   \csgdef{glo@\glsdetoklabel{##1}@prevcount}{##2}%
3828 }%
3829 \let\glsenableentrycount\relax
3830 \renewcommand*{\glsenableentryunitcount}{%
3831   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space
3832     can't be used with \string\glsenableentrycount}%
3833   {Use one or other but not both commands}%
3834 }%
3835 }
```

`\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.

```
3836 \renewcommand*{\@gls@write@entrycounts}{%
3837   \immediate\write\@auxout
3838   {\string\providecommand*{\string\@gls@entry@count}[2]{}}%
3839   \count@=0\relax
3840   \forallglsentries{\@glsentry}{%
3841     \glshasattribute{\@glsentry}{entrycount}%
3842     {%
3843       \ifglsused{\@glsentry}%
3844       {%
3845         \immediate\write\@auxout
3846         {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
3847       }%
3848     }%
3849     \advance\count@ by \@ne
3850   }%
3851 }%
3852 }%
3853 \ifnum\count@=0
3854   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3855     \MessageBreak with \string\glsenableentrycount\space but the
3856     \MessageBreak attribute 'entrycount' hasn't
3857     \MessageBreak been assigned to any of the defined
3858     \MessageBreak entries}%
3859 \fi
3860 }
```

`\trifcounttrigger`

<code>\glstrifcounttrigger{<label>}{<trigger format>}{<normal>}</code>
--

```

3861 \newcommand*{\glxtrifcounttrigger}[3]{%
3862   \glshasattribute{#1}{entrycount}%
3863   {%
3864     \ifnum\gl Sentryprevcount{#1}>\gl sgetattribute{#1}{entrycount}\relax
3865     #3%
3866   \else
3867     #2%
3868   \fi
3869 }%
3870 {#3}%
3871 }

```

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

\@@cgl@

```

3872 \def\@@cgl@#1#2[#3]{%
3873   \glxtrifcounttrigger{#2}%
3874   {%
3875     \cgl sformat{#2}{#3}%
3876     \gl sunset{#2}%
3877   }%
3878   {%
3879     \@gl s@{#1}{#2}[#3]%
3880   }%
3881 }%

```

\@@cgl spl@

```

3882 \def\@@cgl spl@#1#2[#3]{%
3883   \glxtrifcounttrigger{#2}%
3884   {%
3885     \cgl splformat{#2}{#3}%
3886     \gl sunset{#2}%
3887   }%
3888   {%
3889     \@gl spl@{#1}{#2}[#3]%
3890   }%
3891 }%

```

\@@cGl@

```

3892 \def\@@cGl@#1#2[#3]{%
3893   \glxtrifcounttrigger{#2}%
3894   {%
3895     \cGl sformat{#2}{#3}%
3896     \gl sunset{#2}%
3897   }%
3898   {%
3899     \@Gl s@{#1}{#2}[#3]%
3900   }%
3901 }%

```

\@@cGlsp1@

```
3902 \def\@@cGlsp1@#1#2[#3]{%
3903   \glxtrifcounttrigger{#2}%
3904   {%
3905     \cGlsp1format{#2}{#3}%
3906     \glset{#2}%
3907   }%
3908   {%
3909     \@Glsp1@{#1}{#2}[#3]%
3910   }%
3911 }%
```

\@@cGLS@

```
3912 \def\@@cGLS@#1#2[#3]{%
3913   \glxtrifcounttrigger{#2}%
3914   {%
3915     \cGLSformat{#2}{#3}%
3916     \glset{#2}%
3917   }%
3918   {%
3919     \@GLS@{#1}{#2}[#3]%
3920   }%
3921 }%
```

\@@cGLSp1@

```
3922 \def\@@cGLSp1@#1#2[#3]{%
3923   \glxtrifcounttrigger{#2}%
3924   {%
3925     \cGLSp1format{#2}{#3}%
3926     \glset{#2}%
3927   }%
3928   {%
3929     \@GLSp1@{#1}{#2}[#3]%
3930   }%
3931 }%
```

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

\@cgl's@

```
3932 \def\@cgl's@#1#2[#3]{\@gl's@{#1}{#2}[#3]}
```

\@cGL's@

```
3933 \def\@cGL's@#1#2[#3]{\@GL's@{#1}{#2}[#3]}
```

\@cgl'spl@

```
3934 \def\@cgl'spl@#1#2[#3]{\@gl'spl@{#1}{#2}[#3]}
```


\@cGLspl@

```
3935 \def\@cGLspl@#1#2[#3]{\@GLspl@{#1}{#2}[#3]}
```

Add all upper case versions not provided by glossaries.

\cGLS

```
3936 \newrobustcmd*{\cGLS}{\@gls@hyp@opt\@cGLS}
```

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

```
3937 \newcommand*{\@cGLS}[2][\%  
3938 \new@ifnextchar[\@cGLS@{#1}{#2}]{\@cGLS@{#1}{#2}[ ]}%  
3939 }
```

\@cGLS@

```
3940 \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.

```
3941 \newcommand*{\cGLSformat}[2]{%  
3942 \expandafter\mfirstucMakeUppercase\expandafter{\cglformat{#1}{#2}}%  
3943 }
```

\cGLSpl

```
3944 \newrobustcmd*{\cGLSpl}{\@gls@hyp@opt\@cGLSpl}
```

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

```
3945 \newcommand*{\@cGLSpl}[2][\%  
3946 \new@ifnextchar[\@cGLSpl@{#1}{#2}]{\@cGLSpl@{#1}{#2}[ ]}%  
3947 }
```

\@cGLSpl@

```
3948 \def\@cGLSpl@#1#2[#3]{\@GLSpl@{#1}{#2}[#3]}
```

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

```
3949 \newcommand*{\cGLSplformat}[2]{%  
3950 \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%  
3951 }
```

Modify the trigger formats to check for the regular attribute.

\cglformat

```
3952 \renewcommand*{\cglformat}[2]{%  
3953 \glsifregular{#1}  
3954 {\glsentryfirst{#1}}%  
3955 {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%  
3956 }
```

\cGlsformat

```
3957 \renewcommand*{\cGlsformat}[2]{%
3958   \glsifregular{#1}
3959   {\Glsentryfirst{#1}}%
3960   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}#2%
3961 }
```

\cglspformat

```
3962 \renewcommand*{\cglspformat}[2]{%
3963   \glsifregular{#1}
3964   {\glsentryfirstplural{#1}}%
3965   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}#2%
3966 }
```

\cGlsplformat

```
3967 \renewcommand*{\cGlsplformat}[2]{%
3968   \glsifregular{#1}
3969   {\Glsentryfirstplural{#1}}%
3970   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2%
3971 }
```

New code similar to above for unit counting.

defunitcounters

```
3972 \newcommand*{\@@newglossaryentry@defunitcounters}{%
3973   \protected@edef\@glo@countunit{\csuse{@glxtr@categoryattr\@@@glo@category @unitcount}}%
3974   \ifdefvoid\@glo@countunit
3975   {}%
3976   {%
3977     \@glxtr@ifunitcounter{\@glo@countunit}%
3978     {}%
3979     {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}%
3980   }%
3981 }
```

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

```
3982 \newcommand*{\@glxtr@unitcountlist}{}
```

@addunitcounter

```
3983 \newcommand*{\@glxtr@addunitcounter}[1]{%
3984   \listadd{\@glxtr@unitcountlist}{#1}%
3985   \ifcsundef{glxtr@theunit@#1}
3986   {%
3987     \ifcsdef{theH#1}%
3988     {\csdef{glxtr@theunit@#1}{\csuse{theH#1}}}%
3989     {\csdef{glxtr@theunit@#1}{\csuse{the#1}}}%
3990   }%
3991   {}%
3992 }
```

r@ifunitcounter

```
3993 \newcommand*{\@glsxtr@ifunitcounter}[3]{%
3994   \xifinlist{#1}{\@glsxtr@unitcountlist}{#2}{#3}%
3995 }
```

urrentunitcount

```
3996 \newcommand*\@glsxtr@currentunitcount[1]{%
3997   glo@\glsdetoklabel{#1}@currunit@\glsgetattribute{#1}{unitcount}.%
3998   \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}%
3999 }
```

eviousunitcount

```
4000 \newcommand*\@glsxtr@previousunitcount[1]{%
4001   glo@\glsdetoklabel{#1}@prevunit@\glsgetattribute{#1}{unitcount}.%
4002   \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}%
4003 }
```

t@currunitcount

```
4004 \newcommand*{\@gls@increment@currunitcount}[1]{%
4005   \glshasattribute{#1}{unitcount}%
4006   {%
4007     \protected@edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
4008     \ifcsundef{\@glsxtr@csname}%
4009     {%
4010       \csgdef{\@glsxtr@csname}{1}%
4011       \listcsxadd
4012       {glo@\glsdetoklabel{#1}@unitlist}%
4013       {\glsgetattribute{#1}{unitcount}.%
4014       \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}%
4015       }%
4016     }%
4017     {%
4018       \csxdef{\@glsxtr@csname}%
4019       {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
4020     }%
4021   }%
4022   {%
4023 }
```

t@currunitcount

```
4024 \newcommand*{\@gls@local@increment@currunitcount}[1]{%
4025   \glshasattribute{#1}{unitcount}%
4026   {%
4027     \protected@edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
4028     \ifcsundef{\@glsxtr@csname}%
4029     {%
4030       \csdef{\@glsxtr@csname}{1}%
4031       \listcseadd
4032       {glo@\glsdetoklabel{#1}@unitlist}%

```

```

4033     {\glsgetattribute{#1}{unitcount}}.%
4034     \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
4035 }%
4036 }%
4037 {%
4038     \csedef{\@glsxtr@csname}%
4039     {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
4040 }%
4041 }%
4042 {}%
4043 }

```

r@currunitcount

```

4044 \newcommand*{\@glsxtr@currunitcount}[2]{%
4045   \ifcsundef
4046   {glo@\glsdetoklabel{#1}@currunit@#2}%
4047   {0}%
4048   {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}}%
4049 }%

```

r@prevunitcount

```

4050 \newcommand*{\@glsxtr@prevunitcount}[2]{%
4051   \ifcsundef
4052   {glo@\glsdetoklabel{#1}@prevunit@#2}%
4053   {0}%
4054   {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}}%
4055 }%

```

eentryunitcount

```

4056 \newcommand*{\glsenableentryunitcount}{%
    Enable new fields:
4057   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
    Just in case the user has switched on the docdef option.
4058   \renewcommand*{\gls@defdocnewglossaryentry}{%
4059     \renewcommand*\newglossaryentry[2]{%
4060       \PackageError{glossaries}{\string\newglossaryentry\space
4061       may only be used in the preamble when entry counting has
4062       been activated}{If you use \string\glsenableentryunitcount\space
4063       you must place all entry definitions in the preamble not in
4064       the document environment}}%
4065     }%
4066   }%
    New commands to access new fields:
4067   \newcommand*{\glsentrycurrcount}[1]{%
4068     \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
4069     \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
4070   }%

```

```

4071 \newcommand*{\glstentryprevcount}[1]{%
4072   \@glstxtr@prevunitcount{##1}{\glstgetattribute{##1}{unitcount}}.%
4073   \csuse{glstxtr@theunit@\glstgetattribute{##1}{unitcount}}}%
4074 }%

Access total count:
4075 \newcommand*{\glstentryprevtotalcount}[1]{%
4076   \ifcsundef{glo@\glstetoklabel{##1}@prevunittotal}%
4077   {0}%
4078   {%
4079     \number\csuse{glo@\glstetoklabel{##1}@prevunittotal}
4080   }%
4081 }%

Access max value:
4082 \newcommand*{\glstentryprevmaxcount}[1]{%
4083   \ifcsundef{glo@\glstetoklabel{##1}@prevunitmax}%
4084   {0}%
4085   {%
4086     \number\csuse{glo@\glstetoklabel{##1}@prevunitmax}
4087   }%
4088 }%

Adjust post unset and reset:
4089 \let\@glstxtr@entryunitcount@org@unset\glstxtrpostunset
4090 \renewcommand*{\glstxtrpostunset}[1]{%
4091   \@glstxtr@entryunitcount@org@unset{##1}%
4092   \@glst@increment@currunitcount{##1}%
4093 }%
4094 \let\@glstxtr@entryunitcount@org@localunset\glstxtrpostlocalunset
4095 \renewcommand*{\glstxtrpostlocalunset}[1]{%
4096   \@glstxtr@entryunitcount@org@localunset{##1}%
4097   \@glst@local@increment@currunitcount{##1}%
4098 }%
4099 \let\@glstxtr@entryunitcount@org@reset\glstxtrpostreset
4100 \renewcommand*{\glstxtrpostreset}[1]{%
4101   \glshasattribute{##1}{unitcount}%
4102   {%
4103     \protected@edef\@glstxtr@cename{\@glstxtr@currentunitcount{##1}}%
4104     \ifcsundef{\@glstxtr@cename}%
4105     {}%
4106     {\csgdef{\@glstxtr@cename}{0}}%
4107   }%
4108   {}%
4109 }%
4110 \let\@glstxtr@entryunitcount@org@localreset\glstxtrpostlocalreset
4111 \renewcommand*{\glstxtrpostlocalreset}[1]{%
4112   \@glstxtr@entryunitcount@org@localreset{##1}%
4113   \glshasattribute{##1}{unitcount}%
4114   {%

```

```

4115     \protected@edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
4116     \ifcsundef{\@glxtr@csname}%
4117     {}%
4118     {\csdef{\@glxtr@csname}{0}}%
4119     }%
4120     {}%
4121     }%

```

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

```

4122 \let\@cgl@\@cgl@
4123 \let\@cgl@spl\@cgl@spl

4124 \let\@cGls@\@cGls@
4125 \let\@cGls@spl\@cGls@spl
4126 \let\@cGLS@\@cGLS@
4127 \let\@cGLS@spl\@cGLS@spl

```

Write information to the aux file.

```

4128 \AtEndDocument{\@gls@write@entryunitcounts}%
4129 \renewcommand*{\@gls@entry@unitcount}[3]{%
4130   \csgdef{glo@glstdetoklabel{##1}@prevunit@##3}{##2}%
4131   \ifcsundef{glo@glstdetoklabel{##1}@prevunittotal}%
4132   {\csgdef{glo@glstdetoklabel{##1}@prevunittotal}{##2}}%
4133   {%
4134     \csxdef{glo@glstdetoklabel{##1}@prevunittotal}{
4135       \number\numexpr\csuse{glo@glstdetoklabel{##1}@prevunittotal}+##2}%
4136     }%
4137     \ifcsundef{glo@glstdetoklabel{##1}@prevunitmax}%
4138     {\csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}}%
4139     {%
4140       \ifnum\csuse{glo@glstdetoklabel{##1}@prevunitmax}<##2
4141       \csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}%
4142       \fi
4143     }%
4144   }%
4145   \let\glsenableentryunitcount\relax
4146   \renewcommand*{\glsenableentrycount}{%
4147     \PackageError{glossaries-extra}{\string\glsenableentrycount\space
4148       can't be used with \string\glsenableentryunitcount}%
4149     {Use one or other but not both commands}%
4150   }%
4151 }
4152 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

4153 \newcommand*{\@gls@entry@unitcount}[3]{%

```

ryunitcounts@do

```

4154 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
4155   \immediate\write\@auxout
4156   {\string\@gls@entry@unitcount
4157    {\@glsentry}%
4158    {\@glsxtr@currunitcount{\@glsentry}{#1}%
4159    }%
4160    {#1}}%
4161 }

```

entryunitcounts

```

4162 \newcommand*{\@gls@write@entryunitcounts}{%
4163   \immediate\write\@auxout
4164   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
4165   \count@=0\relax
4166   \forallglsentries{\@glsentry}{%
4167     \glshasattribute{\@glsentry}{unitcount}%
4168     {%
4169       \ifglsused{\@glsentry}%
4170       {%
4171         \forlistcsloop
4172           {\@gls@write@entryunitcounts@do}%
4173           {glo@\glsdetoklabel{\@glsentry}@unitlist}%
4174         }%
4175       }%
4176       \advance\count@ by \@ne
4177     }%
4178   }%
4179 }%
4180 \ifnum\count@=0
4181   \GlossariesExtraWarningNoLine{Entry counting has been enabled
4182   \MessageBreak with \string\glsenableentryunitcount\space but the
4183   \MessageBreak attribute ‘unitcount’ hasn’t
4184   \MessageBreak been assigned to any of the defined
4185   \MessageBreak entries}%
4186 \fi
4187 }

```

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.

```

4188 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

4189   \glsenableentryunitcount

```

Redefine \gls etc:

```

4190   \renewcommand*{\gls}{\cgl}%
4191   \renewcommand*{\Gls}{\cGls}%
4192   \renewcommand*{\glspl}{\cglspl}%
4193   \renewcommand*{\Glspl}{\cGlspl}%
4194   \renewcommand*{\GLS}{\cGLS}%

```

```
4195 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
4196 \@glstr@setentryunitcountunsetattr{#1}{#2}{#3}%
```

In case this command is used again:

```
4197 \let\GlsXtrEnableEntryUnitCounting\@glstr@setentryunitcountunsetattr
4198 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
4199 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
4200 can't be used with \string\GlsXtrEnableEntryUnitCounting}%
4201 {Use one or other but not both commands}}%
4202 }
```

tcountunsetattr

```
4203 \newcommand*{\@glstr@setentryunitcountunsetattr}[3]{%
4204 \@for\@glstr@cat:=#1\do
4205 {%
4206 \ifdefempty{\@glstr@cat}{}%
4207 {%
4208 \glsssetcategoryattribute{\@glstr@cat}{entrycount}{#2}%
4209 \glsssetcategoryattribute{\@glstr@cat}{unitcount}{#3}%
4210 }%
4211 }%
4212 }
```

1.3.6 Acronym Modifications

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

nericNewAcronym

```
4213 \renewcommand*{\SetGenericNewAcronym}{%
```

Make sure `\RestoreAcronyms` has been used.

```
4214 \ifdefequal\@addtoacronymlists\@glstr@org@addtoacronymlists
4215 {}%
4216 {%
4217 \GlossariesWarning{\string\SetGenericNewAcronym\space used
4218 without restoring base acronym functions with
4219 \string\RestoreAcronyms}%
4220 }%
4221 \let\@Gls@entryname\@Gls@acrenryname
```

Redefine `\newacronym`:

```
4222 \renewcommand{\newacronym}[4][[]]{%
4223 \ifdefempty{\@glssacronymlists}%
```



```

4224 {%
4225     \def\@glo@type{\acronymtype}%
4226     \setkeys{glossentry}{##1}%
4227     \DeclareAcronymList{\@glo@type}%
4228 }%
4229 {}%
4230 \glskeylisttok{##1}%
4231 \glslabeltok{##2}%
4232 \glsshorttok{##3}%
4233 \glslongtok{##4}%
4234 \newacronymhook
4235 \protected@edef\@do@newglossaryentry{%
4236     \noexpand\newglossaryentry{\the\glslabeltok}%
4237     {%
4238         type=\acronymtype,%
4239         name={\expandonce{\acronymentry{##2}}},%
4240         sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
4241         text={\the\glsshorttok},%
4242         short={\the\glsshorttok},%
4243         shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4244         long={\the\glslongtok},%
4245         longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4246         category=acronym,
4247         \GenericAcronymFields,%
4248         \the\glskeylisttok
4249     }%
4250 }%
4251 \@do@newglossaryentry
4252 }%
4253 \renewcommand*{\acrfullfmt}[3]{%
4254     \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
4255 \renewcommand*{\Acrfullfmt}[3]{%
4256     \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
4257 \renewcommand*{\ACRfullfmt}[3]{%
4258     \glslink[##1]{##2}{%
4259         \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
4260 \renewcommand*{\acrfullplfmt}[3]{%
4261     \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
4262 \renewcommand*{\Acrfullplfmt}[3]{%
4263     \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
4264 \renewcommand*{\ACRfullplfmt}[3]{%
4265     \glslink[##1]{##2}{%
4266         \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
4267 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
4268 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
4269 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
4270 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
4271 }

```

This will cause a problem for glossaries that contain a mixture of acronyms and abbrevia-

tions, so redefine `\newacronym` to use the new abbreviation interface.

First save the original definitions:

```
4272 \let\@glxtr@org@setacronymstyle\setacronymstyle
4273 \let\@glxtr@org@newacronymstyle\newacronymstyle
```

Save the list of acronyms in case they are required.

`tr@acronymlists`

```
4274 \let\@glxtr@acronymlists\@glxacronymlists
```

`dtoacronymlists`

```
4275 \let\@glxtr@org@addtoacronymlists\@addtoacronymlists
```

`setacronymlists`

```
4276 \let\@glxtr@org@setacronymlists\SetAcronymLists
```

Need to provide a replacement for `\forallacronyms` since `\@glxacronymlists` isn't available.

`lsxtr@abbrlists`

```
4277 \newcommand{\@glxtr@abbrlists}{}
```

`breviationlists`

```
4278 \newcommand*{\forallabbreviationlists}[2]{%
4279   \for#1:=\@glxtr@abbrlists\do{\ifdefempty{#1}{-}{#2}}%
4280 }
```

`bbreviationlist`

```
4281 \newcommand*{\@glxtr@addabbreviationlist}[1]{%
4282   \protected@edef\@glo@type{#1}%
4283   \ifdefempty\@glxtr@abbrlists
4284     {\let\@glxtr@abbrlists\@glo@type}%
4285   {%
4286     \ifdefequal\@glxtr@abbrlists\@glo@type
4287     }%
4288   {%

4289     \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@abbrlists}{-}%
4290     {\protected@eappto\@glxtr@abbrlists{,\@glo@type}}%
4291   }%
4292 }%
4293 }
```

`\forallacronyms` Modify to add warning.

```
4294 \renewcommand*{\forallacronyms}[2]{%
4295   \@glxtr@base@acrcmd\forallacronyms\forallabbreviationlists
4296   \for#1:=\@glxacronymlists\do{\ifx#1\@empty\else#2\fi}%
4297 }
```

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

```
4298 \newcommand*{\MakeAcronymsAbbreviations}{%
```

Undo acronym display style:

```
4299 \@for\@gls@type:=\@glsacronymlists\do{%
4300 \csgdef{gls@\@gls@type @entryfmt}{\glsentryfmt}%
4301 }%
```

Save and clear acronym list.

```
4302 \let\@glsxtr@acronymlists\@glsacronymlists
4303 \let\@glsacronymlists\@empty
4304 \let\@addtoacronymlists\@gobble
4305 \let\SetAcronymLists\@gobble
```

Warn if `\acrshort` etc are used.

```
4306 \let\@glsxtr@base@acrcmd\@@glsxtr@base@acrcmd@warn
```

Redefine `\newacronym` to use same interface as `\newabbreviation`.

```
4307 \renewcommand*{\newacronym}[4][{}]{%
4308 \glsxtr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
4309 }%
4310 \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
4311 \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%
4312 \renewcommand*{\setacronymstyle}[1]{%
4313 \PackageError{glossaries-extra}{\string\setacronymstyle{##1}
4314 unavailable.
4315 Use \string\setabbreviationstyle[acronym]\space instead.
4316 The original acronym interface can be restored with
4317 \string\RestoreAcronyms}{}%
4318 }%
4319 \renewcommand*{\newacronymstyle}[1]{%
4320 \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
4321 available unless you restore the original acronym interface with
4322 \string\RestoreAcronyms}%
4323 \@glsxtr@org@newacronymstyle{##1}%
4324 }%
4325 }
```

Switch acronyms to abbreviations:

```
4326 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```
4327 \newcommand*{\RestoreAcronyms}{%
```

Restore acronym list.

```
4328 \let\@glsacronymlists\@glsxtr@acronymlists
4329 \let\@addtoacronymlists\@glsxtr@org@addtoacronymlists
4330 \let\SetAcronymLists\@glsxtr@org@setacronymlists
```

Suppress warnings if \acrshort etc are used.

```
4331 \let\@glxtr@base@acrcmd\@gobbletwo
```

Restore acronym display style:

```
4332 \@for\@gls@type:=\@glsacronymlists\do{%
4333   \SetDefaultAcronymDisplayStyle{\@gls@type}%
4334 }%
```

Switch to the generic acronym mechanism.

```
4335 \SetGenericNewAcronym
4336 \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
4337 \renewcommand{\acronymfont}[1]{##1}%
4338 \let\setacronymstyle\@glxtr@org@setacronymstyle
4339 \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.

```
4340 \renewcommand*\@gls@link@checkfirsthyper{%
4341   \ifglused{\glslabel}%
4342   {\let\glxtrifwasfirstuse\@secondoftwo}
4343   {\let\glxtrifwasfirstuse\@firstoftwo}%
4344   \@glxtr@org@checkfirsthyper
4345 }
4346 \glssetcategoryattribute{acronym}{regular}{false}%
4347 \setacronymstyle{long-short}%
4348 }
```

\glsacspace Allow the user to customise the maximum value.

```
4349 \renewcommand*\glsacspace[1]{%
4350   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
4351   \ifdim\dimen@<\glsacspacemax~\else\space\fi
4352 }
```

\glsacspacemax Value used in the above.

```
4353 \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

```
4354 \newcommand*\@glxtr@reg@glosslist{}
```

Save the original definition of \makeglossaries:

```
4355 \let\@glxtr@org@makeglossaries\makeglossaries
```

`noprintglossary` This command was only introduced to glossaries v4.47 so it may not be defined.

```
4356 \providecommand\@makeglossaries@warn@noprintglossary{%
4357   \ifdefstring{\@glo@types}{,}%
4358   {%
4359     \GlossariesWarningNoLine{No glossaries have been defined}%
4360   }%
4361   {%
4362     \GlossariesWarningNoLine{No \string\printglossary\space
4363       or \string\printglossaries\space
4364       found. ^^J(Remove \string\makeglossaries\space if you
4365       don't want any glossaries.) ^^JThis document will not
4366       have a glossary}%
4367   }%
4368 }%
4369 % \begin{macrocode}
4370 %\end{macro}
4371 %
4372 %\begin{macro}{\@domakeglossaries}
4373 %\changes{1.42}{2020-02-03}{provided definition for \cs{@domakeglossaries}}
4374 % \sty{glossaries} v4.45 introduced \cs{@domakeglossaries} to
4375 % provide a way of disabling \cs{makeglossaries}. If it hasn't been
4376 % defined, define here to do its argument:
4377 % \begin{macrocode}
4378 \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`

```
4379 \renewcommand*{\makeglossaries}[1][]{%
4380   \@domakeglossaries
4381   {%
4382     \@glxtr@if@record@only
4383     {%
4384       \PackageError{glossaries-extra}{\string\makeglossaries\space
4385         not permitted\MessageBreak with record=\@glxtr@record@setting\space
4386         package option}%
4387       {You may only use \string\makeglossaries\space with
4388         record=off or record=hybrid options}%
4389     }%
4390     {%
4391       \ifblank{#1}%
4392       {%
4393         \@glxtr@org@makeglossaries
4394
4395         \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
4396         \let\warn@noprintglossary\@glxtr@warn@hybrid@noprintgloss
```

```

4396     \fi
4397   }%
4398   {%
4399     \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
4400       \PackageError{glossaries-extra}{\string\makeglossaries[#1]\space
4401         not permitted\MessageBreak with record=\@glxtr@record@setting\space package option}%
4402       {You may only use the hybrid \string\makeglossaries[...]\space with
4403         record=off option}%
4404     \else

      \@gls@@automake@immediate was introduced to glossaries v4.42 so it may not be defined.
4405     \ifdef\@gls@@automake@immediate{\@gls@@automake@immediate}{}%
4406     \protected@edef\@glxtr@reg@glosslist{#1}%
4407     \ifundef\glswrite{\newwrite\glswrite}{}%
4408     \protected@write\@auxout{}\string\providecommand
4409       \string\@glsorder[1]{%
4410     \protected@write\@auxout{}\string\providecommand
4411       \string\@istfilename[1]{%
4412     \protected@write\@auxout{}\string\@istfilename{\istfilename}}%
4413     \protected@write\@auxout{}\string\@glsorder{\glsorder}%
4414     \protected@write\@auxout{}\string\glxtr@makeglossaries{#1}%
4415     \write\@auxout{\string\providecommand\string\@gls@reference[3]{}%

      Iterate through each supplied glossary type and activate it.
4416     \@for\@glo@type:=#1\do{%
4417       \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
4418     }%

      New glossaries must be created before \makeglossaries:
4419     \renewcommand*\newglossary[4][{}]{%
4420     \PackageError{glossaries}{New glossaries
4421     must be created before \string\makeglossaries}{You need
4422     to move \string\makeglossaries\space after all your
4423     \string\newglossary\space commands}}%

      Any subsequence instances of this command should have no effect.
4424     \let\@makeglossary\gobble

      Version 1.42 removed letting \makeglossary to \relax (no kernel redefs may be in effect).
4425     \renewcommand\makeglossaries[1][{}]{%

      Disable all commands that have no effect after \makeglossaries
4426     \@disable@onlypremakeg

      Allow see key:
4427     \let\gls@checkseeallowed\relax

      Adjust \@do@seeglossary. This needs to check for the entry's existence but don't increment
      associated counter.
4428     \renewcommand*\@do@seeglossary[2]{%
4429       \glsdoifexists{##1}%
4430     }%

```

```

4431         \protected@edef\@gls@label{\glsdetoklabel{##1}}%
4432         \protected@edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
4433         \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
4434         {\@glsxtr@org@doseeglossary{##1}{##2}}%
4435         {%
4436         \@@glsxtrwrglossmark
4437         \protected@write\@auxout{%
4438         \string\@gls@reference
4439         {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
4440         }%
4441         }%
4442     }%
4443 }%

```

Adjust \@do@@wrglossary

```

4444     \let\@glsxtr@@do@@wrglossary\@@do@@wrglossary
4445     \def\@@do@@wrglossary{%
4446
4447         \protected@edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
4448         \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
4449         {\@glsxtr@@do@@wrglossary}%
4450         {\@gls@noidxglossary}%
4451     }%

```

Suppress warning about no \makeglossaries

```

4451     \let\warn@nomakeglossaries\relax
4452     \let\warn@noprintglossary\@makeglossaries@warn@noprintglossary

```

Only warn for glossaries not listed.

```

4453     \renewcommand{\@gls@noref@warn}[1]{%
4454
4455         \protected@edef\@gls@type{##1}%
4456         \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
4457         {%
4458         \GlossariesExtraWarning{Can't use
4459         \string\printnoidxglossary[type={\@gls@type}]
4460         when '\@gls@type' is listed in the optional argument of
4461         \string\makeglossaries}%
4462         }%
4463         {%
4464         \GlossariesWarning{Empty glossary for
4465         \string\printnoidxglossary[type={##1}].
4466         Rerun may be required (or you may have forgotten to use
4467         commands like \string\gls)}%
4468         }%

```

Adjust display number list to check for type:

```

4469     \renewcommand*{\glsdisplaynumberlist}[1]{%
4470     \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
4471     {\@glsxtr@idx@displaynumberlist{##1}}%

```

```

4472         {\@glxtr@noidx@displaynumberlist{##1}}}%
4473     }%

```

Adjust entry list:

```

4474     \renewcommand*{\glsentrynumberlist}[1]{%
4475         \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
4476         {\@glxtr@idx@entrynumberlist{##1}}}%
4477     {\@glxtr@noidx@entrynumberlist{##1}}}%
4478 }%

```

Adjust number list loop

```

4479     \renewcommand*{\glsnumberlistloop}[2]{%
4480         \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
4481         {%
4482             \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
4483                 not available for glossary ‘##1’}{}%
4484         }%
4485         {\@glxtr@noidx@numberlistloop{##1}{##2}}}%
4486     }%

```

Only sanitize sort for normal indexing glossaries.

```

4487     \renewcommand*{\glsprestandardsort}[3]{%
4488         \expandafter\DTLifinlist\expandafter{##2}{\@glxtr@reg@glosslist}%
4489         {%
4490             \glsdosanitizesort
4491         }%
4492         {%
4493             \ifglssanitizesort
4494                 \@gls@noidx@sanitizesort
4495             \else
4496                 \@gls@noidx@nosanitizesort
4497             \fi
4498         }%
4499     }%

```

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

```

4500     \renewcommand*{\new@glossaryentry}[2]{%
4501         \PackageError{glossaries-extra}{Glossary entries must be defined
4502             in the preamble\MessageBreak when you use the optional argument
4503             of \string\makeglossaries}{Either move your definitions to the
4504             preamble or don't use the optional argument of
4505             \string\makeglossaries}%
4506     }%

```

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).

```

4507     \let\@glo@assign@sortkey\@glxtr@mixed@assign@sortkey
4508     \renewcommand*{\@printgloss@setsort}{%

```

Need to extract just the type value.


```

4509      \expandafter\@glstr@gettype\expandafter,\@glstr@printglossopts,%
4510      type=\glsdefaulttype,\@end@glstr@gettype
4511      \def\@glo@sorttype{\@glo@default@sorttype}%
4512  }%

```

Check automake setting:

```

4513      \ifglautomake
4514      \renewcommand*{\@gls@doautomake}{%
4515      \@for\@gls@type:=\@glstr@reg@glosslist\do{%
4516      \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
4517      }%
4518      }%
4519  \fi

```

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

```

4520      \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
4521  \fi
4522 }%
4523 }%
4524 }%
4525 }

```

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 \TeX 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`

```

4526 \ifdef\@printgloss@checkexists
4527 {\newcommand{\glstr@printgloss@checkexists}{\@printgloss@checkexists}}
4528 {\newcommand{\glstr@printgloss@checkexists}[2]{#2}}

```

`rgprintglossary` (This command is also used for on-the-fly setting.)

```

4529 \newcommand{\@glstr@orgprintglossary}[2]{%
4530   \def\@glo@type{\glsdefaulttype}%

```

Add check here.

```

4531   \def\glossarytitle{%
4532     \ifcsdef{\@glo@type\@glo@type @title}%

```

```

4533     {\csuse{@glo@type@{@glo@type @title}}}%
4534     {\glossaryname}}}%
4535 \def\glossarytoctitle{\glossarytitle}%
4536 \let\org@glossarytitle\glossarytitle
4537 \def\@glossarystyle{%
4538   \ifx\@glossary@default@style\relax
4539     \GlossariesWarning{No default glossary style provided \MessageBreak
4540       for the glossary '@glo@type'. \MessageBreak
4541       Using deprecated fallback. \MessageBreak
4542       To fix this set the style with \MessageBreak
4543       \string\setglossarystyle\space or use the \MessageBreak
4544       style key=value option}%
4545   \fi
4546 }%
4547 \def\gls@dotoc@title{\gls@set@toc@title{@glo@type}}%
4548 \let\@org@glossaryentrynumbers\glossaryentrynumbers
4549 \bgroup
4550   \@printgloss@setsort
4551   \setkeys{printgloss}{#1}%
4552   \ifx\glossarytitle\org@glossarytitle
4553   \else
4554     \cslet{@glo@type@{@glo@type @title}}{\glossarytitle}%
4555   \fi
4556   \let\currentglossary\@glo@type
4557   \let\org@glossaryentrynumbers\glossaryentrynumbers
4558   \let\glsnonextpages\@glsnonextpages
4559   \let\glsnextpages\@glsnextpages

4560   \glsxtractivatenopost
4561   \gls@dotoc@title
4562   \@glossarystyle
4563   \let\gls@org@glossaryentryfield\glossentry
4564   \let\gls@org@glossarysubentryfield\subglossentry
4565   \renewcommand{\glossentry}[1]{%
4566     \protected@xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
4567     \gls@org@glossaryentryfield{##1}%
4568   }%
4569   \renewcommand{\subglossentry}[2]{%
4570     \protected@xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4571     \gls@org@glossarysubentryfield{##1}{##2}%
4572   }%
4573   \@gls@preglossaryhook
4574   \glsxtr@printgloss@checkexists{@glo@type}{#2}%
4575 \egroup
4576 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
4577 \global\let\warn@noprintglossary\relax
4578 }

```

ractivatenopost Change \nopostdesc and \glsxtrnopostpunc to behave as they do in the glossary.

```

4579 \newcommand*{\glsxtractivatenopost}{%

```

```

4580 \let\nopostdesc\@nopostdesc
4581 \let\glxtrnopostpunc\@glxtrnopostpunc
4582 }

lsxtrnopostpunc
4583 \newrobustcmd*{\glxtrnopostpunc}{%

sxtrnopostpunc Provide a command that works like \nopostdesc but only switches of the punctuation with-
out suppressing the post-description hook.
4584 \newcommand{\@glxtrnopostpunc}{%
4585 \let\@glxtrnopostpunc\@glxtrnopostpunc
4586 \ifglxtrnopostdot
4587 \renewcommand{\glxtrnopostpunc}{%
4588 \glxtrnopostdottrue
4589 \let\glxtrnopostpunc\@glxtrnopostpunc
4590 \let\glxtrnopostpunc\@glxtrnopostpunc
4591 \glxtrnopostpunc
4592 \glxtrnopostpunc\@glxtrnopostpunc}%
4593 \else
4594 \renewcommand{\glxtrnopostpunc}{%
4595 \let\glxtrnopostpunc\@glxtrnopostpunc
4596 \let\glxtrnopostpunc\@glxtrnopostpunc
4597 \glxtrnopostpunc
4598 \glxtrnopostpunc\@glxtrnopostpunc}%
4599 \fi
4600 \glxtrnopostdotfalse
4601 }

stpnopostpunc
4602 \newcommand*{\@glxtrnopostpunc\@glxtrnopostpunc}{%

estorepostpunc
4603 \newcommand*{\@glxtrrestorepostpunc}{%
4604 \def\@glxtrrestorepostpunc{%
4605 \glxtrrestorepostpunc
4606 \let\@glxtrrestorepostpunc\@empty
4607 \let\@glxtrrestorepostpunc\@empty
4608 }%
4609 }

restorepostpunc Does nothing outside of glossary.
4610 \newcommand*{\glxtrrestorepostpunc}{%

\@printglossary Redefine.
4611 \renewcommand{\@printglossary}[2]{%
4612 \def\@glxtrprintglossary{#1}%
4613 \glxtrprintglossary{#1}{#2}%
4614 }

```

Add a key that switches off the entry targets:

```

4615 \define@choicekey{printgloss}{target}
4616 [\@glxtr@printglossval\@glxtr@printglossnr]%
4617 {true,false}[true]%
4618 {%
4619   \ifcase\@glxtr@printglossnr
4620     \def\@glstarget{\@glsdohypertarget}%
4621   \else
4622     \let\@glstarget\@secondoftwo
4623   \fi
4624 }
```

hypernameprefix

```

4625 \newcommand{\@glxtrhypernameprefix}{{}
```

New to v1.20:

```

4626 \define@key{printgloss}{targetnameprefix}{%
4627   \renewcommand{\@glxtrhypernameprefix}{#1}%
4628 }

4629 \define@key{printgloss}{prefix}{%
4630   \renewcommand{\@glolinkprefix}{#1}%
4631 }

4632 \define@key{printgloss}{label}{%
4633   \glxtrsetglossarylabel{#1}%
4634 }
```

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.

```

4635 \newcommand{\glxtrsetglossarylabel}[1]{%
4636   \renewcommand*{\@glossaryseclabel}{%
4637     \protected@edef\@currentlabelname{\glossarytoctitle}%
4638     \label{#1}%
4639   }%
4640 }
```

xtr@leveloffset

```

4641 \newcount\@glxtr@leveloffset
```

New to v1.44:

```

4642 \define@key{printgloss}{leveloffset}{%
4643   \@glxtr@assign@leveloffset#1\relax
4644 }
```

ign@leveloffset

```

4645 \newcommand*{\@glxtr@assign@leveloffset}{%
4646   \@ifnextchar+{\p@glxtr@assign@leveloffset}{\np@glxtr@assign@leveloffset}%
4647 }
```

ign@leveloffset Discard initial + character.

```
4648 \newcommand*{\p@glxtr@assign@leveloffset}[1]{%
4649 \ifnextchar+{\pp@glxtr@assign@leveloffset}{\np@glxtr@assign@leveloffset}%
4650 }
```

ign@leveloffset

```
4651 \def\np@glxtr@assign@leveloffset#1\relax{\@glxtr@leveloffset=#1\relax}
```

ign@leveloffset

```
4652 \def\pp@glxtr@assign@leveloffset#1\relax{\advance\@glxtr@leveloffset by #1\relax}
```

```
4653 \define@boolkey{printgloss}[glxtr@printgloss@]{groups}[true]{}
4654 \glxtr@printgloss@groupstrue
```

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

```
4655 \let\@glxtr@org@glsdohypertarget\glsdohypertarget
4656 \renewcommand{\glsdohypertarget}[2]{%
4657 \@glxtr@org@glsdohypertarget{\@glxtrhypernameprefix#1}{#2}%
4658 }
```

Update \@glstarget to use \def instead being assigned with \let so that it can pick up the new definition and allow any further redefinitions:

```
4659 \ifx\@glstarget\@glxtr@org@glsdohypertarget
4660 \def\@glstarget{\glsdohypertarget}%
4661 \fi
```

@makeglossaries For the benefit of makeglossaries

```
4662 \newcommand*{\glxtr@makeglossaries}[1]{}%
```

@glxtr@gettype Get just the type.

```
4663 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
4664 \def\@glo@type{#2}%
4665 }
```

@assign@sortkey Assign the sort key.

```
4666 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
4667 \protected@edef\@glo@type{\@glo@type}%
4668 \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
4669 {%
4670 \@glo@no@assign@sortkey{#1}%
4671 }%
4672 {%
4673 \@@glo@assign@sortkey{#1}%
4674 }%
4675 }%
```

Display number list for the regular version:

splaynumberlist

```
4676 \let\@glstr@idx@displaynumberlist\glstrdisplaynumberlist
```

Display number list for the “noidx” version:

splaynumberlist

```
4677 \newcommand*{\@glstr@noidx@displaynumberlist}[1]{%
4678   \letcs{\@glstr@loclist}{glo@\glstrdetoklabel{#1}@loclist}%
4679   \ifdef\@glstr@loclist
4680   {%
4681     \def\@glstr@noidx@loclist@sep{%
4682       \def\@glstr@noidx@loclist@sep{%
4683         \def\@glstr@noidx@loclist@sep{%
4684           \glstrnumlistsep
4685         }%
4686         \def\@glstr@noidx@loclist@finalsep{\glstrnumlistlastsep}%
4687       }%
4688     }%
4689     \def\@glstr@noidx@loclist@finalsep{}%
4690     \def\@glstr@noidx@loclist@prev{}%
4691     \forlistloop{\glstrnoidxdisplayloclisthandler}{\@glstr@loclist}%
4692     \@glstr@noidx@loclist@finalsep
4693     \@glstr@noidx@loclist@prev
4694   }%
4695   {%

4696   \glstrtrundeftag
4697   \glstrdoifexists{#1}%
4698   {%
4699     \GlossariesWarning{Missing location list for ‘#1’. Either
4700       a rerun is required or you haven’t referenced the entry.}%
4701   }%
4702 }%
4703 }%
4704
```

And for the number list loop:

@numberlistloop

```
4705 \newcommand*{\@glstr@noidx@numberlistloop}[3]{%
4706   \letcs{\@glstr@loclist}{glo@\glstrdetoklabel{#1}@loclist}%
4707   \let\@glstr@org@glstrnoidxdisplayloc\glstrnoidxdisplayloc
4708   \let\@glstr@org@glstrseeformat\glstrseeformat
4709   \let\glstrnoidxdisplayloc#2\relax
4710   \let\glstrseeformat#3\relax
4711   \ifdef\@glstr@loclist
4712   {%
4713     \forlistloop{\glstrnoidxnumberlistloophandler}{\@glstr@loclist}%
4714   }%
4715   {%
```

```

4716 \glxtrundeftag
4717 \glsdoifexists{#1}%
4718 {%
4719 \GlossariesWarning{Missing location list for ‘##1’. Either
4720 a rerun is required or you haven’t referenced the entry.}%
4721 }%
4722 }%
4723 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
4724 \let\glsseeformat\@gls@org@glsseeformat
4725 }%

```

Same for entry number list.

entrynumberlist

```

4726 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
4727 \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
4728 \ifdef\@gls@loclist
4729 {%
4730 \glsnoidxloclist{\@gls@loclist}%
4731 }%
4732 {%
4733 \glxtrundeftag
4734 \glsdoifexists{#1}%
4735 {%
4736 \GlossariesWarning{Missing location list for ‘#1’. Either
4737 a rerun is required or you haven’t referenced the entry.}%
4738 }%
4739 }%
4740 }%

```

entrynumberlist

```

4741 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

4742 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
4743 \protected@edef\@glsxtr@titlelabel{#1}%
4744 \ifdefvoid\@glsxtr@titlelabel
4745 {}%
4746 {%
4747 \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle@#1}}%
4748 }%
4749 \ifdefvoid{\@glsxtr@titlelabel}%
4750 {%
4751 \DTLifint{#1}%
4752 {%
4753 \ifnum#1<256\relax
4754 \edef#2{\char#1\relax}%
4755 \else
4756 \edef#2{#1}%

```

```

4757     \fi
4758   }%
4759   {%
4760     \ifcsundef{#1groupname}%
4761     {\def#2{#1}}%
4762     {\letcs#2{#1groupname}}%
4763   }%
4764 }%
4765 {%
4766   \let#2\@glsxtr@titlelabel
4767 }%
4768 }

```

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

```

4769 \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.

```

4770 \newrobustcmd{\glsxtrgetgrouptitle}[2]{%
4771   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
4772   \@onelevel@sanitize\@glsxtr@titlelabel
4773   \ifcsdef{\@glsxtr@titlelabel}
4774   {\letcs{#2}{\@glsxtr@titlelabel}}%
4775   {\glsxtr@org@getgrouptitle{#1}{#2}}%
4776 }
4777 \let\@gls@getgrouptitle\glsxtrgetgrouptitle

```

trsetgrouptitle Sets the title for the given group label.

```

4778 \newcommand{\glsxtrsetgrouptitle}[2]{%
4779   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
4780   \@onelevel@sanitize\@glsxtr@titlelabel
4781   \protected@csxdef{\@glsxtr@titlelabel}{#2}%
4782 }

```

alsetgrouptitle As above put only locally defines the title.

```

4783 \newcommand{\glsxtrlocalsetgrouptitle}[2]{%
4784   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
4785   \@onelevel@sanitize\@glsxtr@titlelabel
4786   \protected@csedef{\@glsxtr@titlelabel}{#2}%
4787 }

```

\glsnavigation Redefine to use new user-level command.

```

4788 \renewcommand*{\glsnavigation}{%
4789   \def\@gls@between{}%
4790   \ifcsundef{\@gls@hypergroup@list@{\@gls@type}}%
4791   {%
4792     \def\@gls@list{}%
4793   }%

```



```

4794 {%
4795   \expandafter\let\expandafter\@gls@list
4796   \csname @gls@hypergroup@list@\@glo@type\endcsname
4797 }%
4798 \@for\@gls@tmp:=\@gls@list\do{%
4799   \@gls@between
4800   \glsxtrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
4801   \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
4802   \let\@gls@between\glshypernavsep
4803 }%
4804 }

```

@noidx@glossary

```

4805 \renewcommand*{\@print@noidx@glossary}{%
4806   \ifcsdef{@glsref@\@glo@type}%
4807   {%
4808     \ifcsdef{@glo@sortmacro@\@glo@sorttype}%
4809     {%
4810       \csuse{@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
4811     }%
4812     {%
4813       \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
4814     }%
4815     \glossarysection[\glossarytoctitle]{\glossarytitle}%
4816     \glossarypreamble

```

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

```

4817   \def\@gls@currentlettergroup{%
4818     \begin{theglossary}%
4819     \glossaryheader
4820     \glsresetentrylist
4821     \forlistcsloop{\@gls@noidx@do}{\@glsref@\@glo@type}%
4822     \end{theglossary}%
4823     \glossarypostamble
4824   }%
4825   {%

```

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

```

4826   \glsxtrifemptyglossary{\@glo@type}%
4827   {}%
4828   {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
4829   \@gls@noref@warn{\@glo@type}%
4830   }%
4831 }

```

noidxdisplayloc Patch to check for range formations.

```

4832 \renewcommand*{\glsnoidxdisplayloc}[4]{%
4833   \setentrycounter{#1}{#2}%

```

```

4834 \@glxstr@display@loc#3\empty\end@glxstr@display@loc{#4}%
4835 }

```

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

```

4836 \def\@glxstr@display@loc#1#2\end@glxstr@display@loc#3{%
4837 \ifx#1\relax
4838 \glxstrdisplaystartloc{#2}{#3}%
4839 \else
4840 \ifx#1)\relax
4841 \glxstrdisplayendloc{#2}{#3}%
4842 \else
4843 \glxstrdisplaysingleloc{#1#2}{#3}%
4844 \fi
4845 \fi
4846 }

```

`isplaysingleloc` Single location.

```

4847 \newcommand*{\glxstrdisplaysingleloc}[2]{%
4848 \csuse{#1}{#2}%
4849 }

```

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 `\glxstrlocrangefmt`.

`displaystartloc` Start of a location range.

```

4850 \newcommand*{\glxstrdisplaystartloc}[2]{%
4851 \protected@edef\glxstrlocrangefmt{#1}%
4852 \ifx\glxstrlocrangefmt\empty
4853 \def\glxstrlocrangefmt{glnumberformat}%
4854 \fi
4855 \expandafter\glxstrdisplaysingleloc
4856 \expandafter{\glxstrlocrangefmt}{#2}%
4857 }

```

`trdisplayendloc` End of a location range.

```

4858 \newcommand*{\glxstrdisplayendloc}[2]{%
4859 \protected@edef\@glxstr@tmp{#1}%
4860 \ifdefempty{\@glxstr@tmp}{\def\@glxstr@tmp{glnumberformat}}}%
4861 \ifx\glxstrlocrangefmt\@glxstr@tmp
4862 \else
4863 \GlossariesExtraWarning{Mismatched end location range
4864 (start=\glxstrlocrangefmt, end=\@glxstr@tmp)}%
4865 \fi
4866 \expandafter\glxstrdisplayendloohook\expandafter{\@glxstr@tmp}{#2}%
4867 \expandafter\glxstrdisplaysingleloc
4868 \expandafter{\glxstrlocrangefmt}{#2}%
4869 \def\glxstrlocrangefmt{}%
4870 }

```

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

```
4871 \newcommand*{\glxtrdisplayendlochook}[2] {}
```

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

```
4872 \newcommand*{\glxtrlocrangefmt} {}
```

setentrycounter Adjust \setentrycounter to save the original prefix.

```
4873 \renewcommand*{\setentrycounter}[2] [] {%
4874   \def\glxtrcounterprefix{#1}%
4875   \ifx\glxtrcounterprefix\@empty
4876     \def\@glo@counterprefix{.}%
4877   \else
4878     \def\@glo@counterprefix{.#1.}%
4879   \fi
4880   \def\glsetentrycounter{#2}%
4881 }
```

ls@removespaces Redefine to allow adjustments to location hyperlink.

```
4882 \def\@gls@removespaces#1 #2\@nil{%
4883   \toks@=\expandafter{\the\toks@#1}%
4884   \ifx\@#2\%
4885     \edef\@glo@tmp{\the\toks@}%
4886     \ifx\@glo@tmp\empty
4887     \else
```

Expand location (just in case \toks@ is needed for something else).

```
4888     \expandafter\glxtrlocationhyperlink\expandafter
4889     \glsetentrycounter\expandafter\@glo@counterprefix\expandafter{\the\toks@}%
4890     \fi
4891   \else
4892     \@gls@ReturnAfterFi{%
4893       \@gls@removespaces#2\@nil
4894     }%
4895   \fi
4896 }
```

cationhyperlink

`\glxtrlocationhyperlink{<counter>}{<prefix>}{<location>}`

```
4897 \newcommand*{\glxtrlocationhyperlink}[3] {%
4898   \ifdefined\glxtrsupplocationurl
4899   {%
4900     \GlsXtrInternalLocationHyperlink{#1}{#2}{#3}%
4901   }%
4902   {%
```

```

4903 \hyperref{\glxtrsupplocationurl}{\#1\#2\#3}{\#3}%
4904 }%
4905 }

```

supphypernumber

```

4906 \newcommand*\glxtrsupphypernumber}[1]{%
4907 {%
4908 \glshasattribute{\glscurrententrylabel}{externallocation}%
4909 {%
4910 \def\glxtrsupplocationurl{%
4911 \glsggetattribute{\glscurrententrylabel}{externallocation}}%
4912 }%
4913 {%
4914 \def\glxtrsupplocationurl{%
4915 }%
4916 \glshypernumber{\#1}%
4917 }%
4918 }

```

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

@print@glossary

```

4919 \renewcommand*\@print@glossary{%
4920 \makeatletter
4921 \@input{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4922 \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4923 {}%
4924 {\glxtrNoGlossaryWarning{\@glo@type}}%
4925 \ifglxindy
4926 \ifcsundef{\xdy@\@glo@type @language}%
4927 {%
4928 \edef\@do@auxoutstuff{%
4929 \noexpand\AtEndDocument{%
4930 \noexpand\immediate\noexpand\write\@auxout{%
4931 \string\providecommand\string\@xdylanguage[2]{}}%
4932 \noexpand\immediate\noexpand\write\@auxout{%
4933 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4934 }%
4935 }%
4936 }%
4937 {%
4938 \edef\@do@auxoutstuff{%
4939 \noexpand\AtEndDocument{%
4940 \noexpand\immediate\noexpand\write\@auxout{%
4941 \string\providecommand\string\@xdylanguage[2]{}}%
4942 \noexpand\immediate\noexpand\write\@auxout{%
4943 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4944 @language\endcsname}}%

```

```

4945     }%
4946 }%
4947 }%
4948 \@do@auxoutstuff
4949 \edef\@do@auxoutstuff{%
4950     \noexpand\AtEndDocument{%
4951         \noexpand\immediate\noexpand\write\@auxout{%
4952             \string\providecommand\string\@gls@codepage[2]{}}%
4953         \noexpand\immediate\noexpand\write\@auxout{%
4954             \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
4955     }%
4956 }%
4957 \@do@auxoutstuff
4958 \fi
4959 \renewcommand*{\@warn@nomakeglossaries}{%
4960     \GlossariesWarningNoLine{\string\makeglossaries\space
4961     hasn't been used,^^Jthe glossaries will not be updated}%
4962 }%
4963 }

```

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

`\GlsWarningHead` Header message.

```

4964 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
4965     This document is incomplete. The external file associated with
4966     the glossary '#1' (which should be called \texttt{#2})
4967     hasn't been created.%
4968 }

```

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

```

4969 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
4970     This has probably happened because there are no entries defined
4971     in this glossary.%
4972 }

```

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

```

4973 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
4974     If you don't want this glossary,
4975     add \texttt{nomain} to your package option list when you load
4976     \texttt{glossaries-extra.sty}. For example:%
4977 }

```

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

```

4978 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
4979     Did you forget to use \texttt{type=#1} when you defined your
4980     entries? If you tried to load entries into this glossary with
4981     \texttt{\string\loadglsentries} did you remember to use
4982     \texttt{[#1]} as the optional argument? If you did, check that
4983     the definitions in the file you loaded all had the type set

```

```

4984 to \texttt{\string\glsdefaulttype}.%
4985 }

```

WarningCheckFile Advisory message to check the file contents.

```

4986 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
4987   Check the contents of the file \texttt{#1}. If
4988   it's empty, that means you haven't indexed any of your entries in this
4989   glossary (using commands like \texttt{\string\gls} or
4990   \texttt{\string\glsadd}) so this list can't be generated.
4991   If the file isn't empty, the document build process hasn't been
4992   completed.%
4993 }

```

WarningAutoMake Message when automake option has been used.

```

4994 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
4995   You may need to rerun \LaTeX. If you already have, it may be that
4996   \TeX's shell escape doesn't allow you to run
4997   \ifglxindy xindy\else makeindex\fi. Check the
4998   transcript file \texttt{\jobname.log}. If the shell escape is
4999   disabled, try one of the following:
5000
5001   \begin{itemize}
5002     \item Run the external (Lua) application:
5003
5004       \texttt{makeglossaries-lite \string"\jobname\string"}
5005
5006     \item Run the external (Perl) application:
5007
5008       \texttt{makeglossaries \string"\jobname\string"}
5009   \end{itemize}
5010
5011   Then rerun \LaTeX\ on this document.
5012   \GlossariesExtraWarning{Rerun required to build the
5013   glossary '#1' or check TeX's shell escape allows
5014   you to run \ifglxindy xindy\else makeindex\fi}%
5015 }

```

WarningMisMatch Mismatching \makenoidxglossaries.

```

5016 \newcommand{\GlsXtrNoGlsWarningMisMatch}{%
5017   You need to either replace \texttt{\string\makenoidxglossaries}
5018   with \texttt{\string\makeglossaries} or replace
5019   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
5020   \texttt{\string\printnoidxglossary}
5021   (or \texttt{\string\printnoidxglossaries}) and then rebuild
5022   this document.%
5023 }

```

WarningBuildInfo Build advice.

```

5024 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
5025   Try one of the following:
5026   \begin{itemize}
5027     \item Add \texttt{automake} to your package option list when you load
5028       \texttt{glossaries-extra.sty}. For example:
5029
5030       \texttt{\string\usepackage[automake]%
5031         \glsopenbrace glossaries-extra\glsclosebrace}
5032
5033     \item Run the external (Lua) application:
5034
5035       \texttt{makeglossaries-lite.lua \string"\jobname\string"}
5036
5037     \item Run the external (Perl) application:
5038
5039       \texttt{makeglossaries \string"\jobname\string"}
5040   \end{itemize}
5041
5042   Then rerun \LaTeX\ on this document.%
5043 }

```

trRecordWarning Paragraph for record=only.

```

5044 \newcommand{\GlsXtrRecordWarning}[1]{%
5045   \texttt{\string\printglossary} doesn't work
5046   with the \texttt{record=@\glxtr@record@setting} package option
5047   use\par\texttt{\string\printunsrtglossary[type=#1]}\par
5048   instead (or change the package option).%
5049 }

```

oGlsWarningTail Final paragraph.

```

5050 \newcommand{\GlsXtrNoGlsWarningTail}{%
5051   This message will be removed once the problem has been fixed.%
5052 }

```

GlsWarningNoOut No out file created. Build advice.

```

5053 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
5054   The file \texttt{#1} doesn't exist. This most likely means you haven't used
5055   \texttt{\string\makeglossaries} or you have used
5056   \texttt{\string\nofiles}. If this is just a draft version of the
5057   document, you can suppress this message using the
5058   \texttt{nomissingglstext} package option.%
5059 }

```

glossarywarning

```

5060 \newcommand*{\@glxtr@defaultnoglossarywarning}[1]{%
5061   \glossarysection[\glossarytoctitle]{\glossarytitle}
5062   \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
5063   \par
5064   \glxtrifemptyglossary{#1}%

```

```

5065 {%
5066     \GlsXtrNoGlsWarningEmptyStart\space
5067     \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
5068     \medskip
5069     \noindent\texttt{\string\usepackage[nomain\ifglsacronym ,acronym\fi]%
5070         \glsopenbrace glossaries-extra\glsclosebrace}
5071     \medskip
5072     }%
5073     {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
5074 }%
5075 {%
5076     \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
5077     {%
5078         \GlsXtrNoGlsWarningCheckFile
5079         {\jobname.\csname @glotype@\@glo@type @out\endcsname}
5080
5081         \ifglsautomake
5082
5083         \GlsXtrNoGlsWarningAutoMake{#1}
5084
5085     \else
5086
5087         \ifthenelse{\equal{#1}{main}}{%
5088             {%
5089                 \GlsXtrNoGlsWarningEmptyMain\par
5090                 \medskip
5091                 \noindent\texttt{\string\usepackage[nomain]%
5092                     \glsopenbrace glossaries-extra\glsclosebrace}
5093                 \medskip
5094             }%
5095             {}%
5096
5097             \ifdefequal\makeglossaries\@no@makeglossaries
5098             {%
5099                 \GlsXtrNoGlsWarningMisMatch
5100             }%
5101             {%
5102                 \GlsXtrNoGlsWarningBuildInfo
5103             }%
5104         \fi
5105     }%
5106     {%
5107         \GlsXtrNoGlsWarningNoOut
5108         {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
5109     }%
5110 }%
5111 \par
5112 \GlsXtrNoGlsWarningTail
5113 }

```


glossarywarning Warn about using `\printglossary` with record

```

5114 \newcommand*{\@glxtr@record@noglossarywarning}[1]{%
5115   \GlossariesExtraWarning{\string\printglossary\space doesn't work\MessageBreak
5116   with record=\@glxtr@record@setting\space package option\MessageBreak(use
5117   \string\printunsrtglossary[type=#1])\MessageBreak
5118   instead (or change the package option)}%
5119   \glossarysection[\glossarytoctitle]{\glossarytitle}
5120   \GlsXtrRecordWarning{#1}
5121   \GlsXtrNoGlsWarningTail
5122 }
```

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

ResourceOptions Default resource options.

```

5123 \newcommand*{\GlsXtrDefaultResourceOptions}{}

```

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

```

5124 \newcommand*{\glxtrresourcefile}[2] []{%

```

The record option can't be set after this command.

```

5125   \disable@keys{glossaries-extra.sty}{record}%
5126   \glxtr@writefields
5127   \ifdefempty\GlsXtrDefaultResourceOptions
5128   {%
5129     \protected@write\@auxout{\glxtrresourceinit}%
5130     {\string\glxtr@resource{#1}{#2}}%
5131   }%
5132   {%
5133     \protected@write\@auxout{\glxtrresourceinit}%
5134     {\string\glxtr@resource{\GlsXtrDefaultResourceOptions,#1}{#2}}%
5135   }%
5136   \let\@glxtr@org@see@noindex\@glxtr@see@noindex
5137   \let\@glxtr@see@noindex\relax
5138   \IfFileExists{#2.glstex}%
5139   {%

```

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

```

5140     \edef\@bibgls@restreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
5141     \makeatletter
5142     \@input{#2.glstex}%
5143     \@bibgls@restreat

```

If the `record=nameref` option has been set, check if this is supported by the installed version of **bib2gls**.

```

5144     \@glxtr@check@bibgls@nameref
5145   }%
5146   {%
5147     \GlossariesExtraWarning{No file '#2.glstex'}%

```

```

5148 }%
5149 \let\@gls@see@noindex\@glsxtr@org@see@noindex
5150 }
5151 \@onlypreamble\glsxtrresourcefile

@bibgls@nameref This will only warn after bib2gls has created the .glstex file, but there's way to check before.
5152 \newcommand{\@glsxtr@check@bibgls@nameref}{%
5153   \ifx\@glsxtr@record@setting\@glsxtr@record@setting@nameref
5154     \ifdef\bibglshrefchar
5155       {}%
5156       {%
5157         \GlossariesExtraWarning{record=nameref requires at least
5158           version 1.8 of bib2gls}%
5159       }%
5160   \fi
5161   \let\@glsxtr@check@bibgls@nameref\relax
5162 }

xtrresourceinit Code used during the protected write operation.
5163 \newcommand*{\glsxtrresourceinit}{%

trresourcecount
5164 \newcount\glsxtrresourcecount

trLoadResources Short cut that uses \glsxtrresourcefile with \jobname as the mandatory argument.
5165 \newcommand*{\GlsXtrLoadResources}[1][{}]{%
5166   \ifnum\glsxtrresourcecount=0\relax
5167     \glsxtrresourcefile[#1]{\jobname}%
5168   \else
5169     \glsxtrresourcefile[#1]{\jobname-\the\glsxtrresourcecount}%
5170   \fi
5171   \advance\glsxtrresourcecount by 1\relax
5172 }

glsxtr@resource
5173 \newcommand*{\glsxtr@resource}[2]{%

\glsxtr@fields
5174 \newcommand*{\glsxtr@fields}[1]{%

xtr@texencoding
5175 \newcommand*{\glsxtr@texencoding}[1]{%

\glsxtr@langtag
5176 \newcommand*{\glsxtr@langtag}[1]{%

@pluralsuffixes
5177 \newcommand*{\glsxtr@pluralsuffixes}[4]{%

```

tr@shortcutsval

```
5178 \newcommand*{\glxtr@shortcutsval}[1]{}
```

lxtr@linkprefix

```
5179 \newcommand*{\glxtr@linkprefix}[1]{}
```

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

```
5180 \newcommand*{\glxtr@writefields}{%
```

```
5181   \protected@write\@auxout{}%
```

```
5182     {\string\providecommand*{\string\glxtr@fields}[1]{}}%
```

```
5183   \protected@write\@auxout{}%
```

```
5184     {\string\providecommand*{\string\glxtr@resource}[2]{}}%
```

```
5185   \protected@write\@auxout{}%
```

```
5186     {\string\providecommand*{\string\glxtr@pluralsuffixes}[4]{}}%
```

```
5187   \protected@write\@auxout{}%
```

```
5188     {\string\providecommand*{\string\glxtr@shortcutsval}[1]{}}%
```

```
5189   \protected@write\@auxout{}%
```

```
5190     {\string\providecommand*{\string\glxtr@linkprefix}[1]{}}%
```

```
5191   \protected@write\@auxout{}{\string\glxtr@fields{\@gls@keymap}}%
```

```
5192   \protected@write\@auxout{}%
```

```
5193     {\string\providecommand*{\string\glxtr@record}[5]{}}%
```

```
5194   \ifx\@glxtr@record@setting\@glxtr@record@setting@nameref
```

```
5195     \protected@write\@auxout{}%
```

```
5196       {\string\providecommand*{\string\glxtr@record@nameref}[8]{}}%
```

```
5197   \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`.

```
5198   \ifdef\CurrentTrackedLanguageTag
```

```
5199     {%
```

```
5200       \protected@write\@auxout{}{%
```

```
5201         \string\glxtr@langtag{\CurrentTrackedLanguageTag}}%
```

```
5202     }%
```

```
5203   {%}
```

```
5204   \protected@write\@auxout{}{\string\glxtr@pluralsuffixes
```

```
5205     {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
```

```
5206     {\glxtrabbrvpluralsuffix}}%
```

```
5207   \ifdef\inputencodingname
```

```
5208     {%
```

```
5209       \protected@write\@auxout{}{\string\glxtr@texencoding{\inputencodingname}}%
```

```
5210     }%
```

```
5211   {%}
```

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.)

```

5212 \ifpackageloaded{fontspec}%
5213 {\protected@write\@auxout{}\string\glsxtr@texencoding{utf8}}}%
5214 {}%
5215 }%
5216 \protected@write\@auxout{}\string\glsxtr@shortcutsval{\glsxtr@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.

```

5217 \AtBeginDocument
5218 {\protected@write\@auxout{}\string\glsxtr@linkprefix{\glsxtr@linkprefix}}}%
5219 \let\glsxtr@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.

```

5220 \ifglsautomake
5221 \IfFileExists{\jobname.aux}%
5222 {\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.

```

5223 \ifx\@gls@doautomake\@gls@doautomake@err
5224 \let\@gls@doautomake\relax
5225 \fi
5226 \fi

```

Check if order=letter has been used by mistake (but not if record=alsoindex has been used).

```

5227 \@glsxtr@if@record@only
5228 {\ifdefstring{\glsorder}{letter}%
5229 {\GlossariesExtraWarningNoLine{Package option 'order=letter' isn't
5230 supported with 'record=\@glsxtr@record@setting'. Use 'break-at=none'
5231 resource option instead}}}%
5232 {}%
5233 }%
5234 {}%
5235 }

```

do@automake@err

```

5236 \newcommand*\@gls@doautomake@err{%
5237 \PackageError{glossaries}{You must use
5238 \string\makeglossaries\space with automake=true}
5239 {%
5240 Either remove the automake=true setting or
5241 add \string\makeglossaries\space to your document preamble.%
5242 }%
5243 }

```

Allow locations specific to a particular counter to be recorded.

`\glxtr@record`

```
5244 \newcommand*{\glxtr@record}[5]{}

@record@nameref  Used with record=nameref to include current label information.
5245 \newcommand*{\glxtr@record@nameref}[8]{}

r@counterrecord  Aux file command.
5246 \newcommand*{\glxtr@counterrecord}[3]{%
5247   \glxtrfieldlistgadd{#1}{record.#2}{#3}%
5248 }

unterrecordhook  Hook used by \@glxtr@dorecord.
5249 \newcommand*{\@glxtr@counterrecordhook}{}

trRecordCounter  Activate recording for a particular counter (identified in the argument).
5250 \newcommand*{\GlsXtrRecordCounter}[1]{%
5251   \@glxtr@recordcounter{#1}%
5252 }
5253 \@onlypreamble\GlsXtrRecordCounter

docounterrecord
5254 \newcommand*{\@glxtr@docounterrecord}[1]{%
5255   \protected@write\@auxout{}\string\glxtr@counterrecord
5256     {\@glx@label}{#1}{\csuse{the#1}}}%
5257 }

lsxtrglossentry  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.
5258 \newcommand*{\glxtrglossentry}[1]{%
5259   \glxtrtitleorpdforheading
5260   {\@glxtrglossentry{#1}}%
5261   {\glsentryname{#1}}%
5262   {\glxtrheadname{#1}}%
5263 }

lsxtrglossentry  Another test is needed in case \@glxtrglossentry has been written to the table of contents.
5264 \newrobustcmd*{\@glxtrglossentry}[1]{%
5265   \glxtrtitleorpdforheading
5266   {%
5267     \glsdoidexists{#1}%
5268     {%
5269       \begingroup
```

```

5270      \protected@edef\glscurrententrylabel{\glsetoklabel{#1}}%
5271      \protected@edef\currentglossary{\GlsXtrStandaloneGlossaryType}%
5272      \ifglshasparent{#1}%
5273      {\GlsXtrStandaloneSubEntryItem{#1}}%
5274      {\glsentryitem{#1}}%
5275      \GlsXtrStandaloneEntryName{#1}%
5276      \endgroup
5277  }%
5278 }%
5279 {\glsentryname{#1}}%
5280 {\glsxtrheadname{#1}}%
5281 }

```

StandaloneEntryName

```

5282 \newcommand*{\GlsXtrStandaloneEntryName}[1]{%
5283   \glstarget{#1}{\glssentryname{#1}}%
5284 }

```

StandaloneGlossaryType To make it easier to adjust the definition of `\currentglossary` within `\glsxtrglossentry`, this expands to the default definition. (If redefined, it must fully expand to the appropriate label.)

```

5285 \newcommand{\GlsXtrStandaloneGlossaryType}{\glsentrytype{\glscurrententrylabel}}

```

StandaloneSubEntryItem Used for sub-entries in standalone format. The argument is the entry's label.

```

5286 \newcommand*{\GlsXtrStandaloneSubEntryItem}[1]{%
5287   \GlsXtrIfFieldEqNum{level}{#1}{1}{\glssubentryitem{#1}}}%
5288 }

```

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.

```

5289 \newcommand*{\glsxtrglossentryother}[3]{%
5290   \ifstrempy{#1}%
5291   {%
5292     \ifcsdef{glsxtrhead#3}%
5293     {%
5294       \glsxtrtitleorpdforheading
5295       {\@glsxtrglossentryother{#2}{#3}{#1}}%
5296       {\@gls@entry@field{#2}{#3}}%
5297       {\csuse{glsxtrhead#3}{#2}}%
5298     }%
5299     {%
5300       \glsxtrtitleorpdforheading
5301       {\@glsxtrglossentryother{#2}{#3}{#1}}%
5302       {\@gls@entry@field{#2}{#3}}%
5303       {\@gls@entry@field{NoCaseChange{#2}}{#3}}%
5304     }%

```

```

5305 }%
5306 {%
5307   \glxtrtitleorpdforheading
5308   {\@glxtrglossentryother{#2}{#3}{#1}}%
5309   {\@gls@entry@field{#2}{#3}}%
5310   {#1}%
5311 }%
5312 }

```

`glossentryother` As `\@glxtrglossentry` but uses a different field.

```

5313 \newrobustcmd*{\@glxtrglossentryother}[3]{%
5314   \glxtrtitleorpdforheading
5315   {%
5316     \glsdoifexists{#1}%
5317     {%
5318       \begingroup
5319         \protected@edef\glscurrententrylabel{\glsetoklabel{#1}}%
5320         \protected@edef\currentglossary{\GlsXtrStandaloneGlossaryType}%
5321         \ifglshasparent{#1}%
5322         {\GlsXtrStandaloneSubEntryItem{#1}}%
5323         {\glsentryitem{#1}}%
5324         \GlsXtrStandaloneEntryOther{#1}%
5325       \endgroup
5326     }%
5327   }%
5328   {\@gls@entry@field{#1}{#2}}%
5329   {#3}%
5330 }

```

`aloneEntryOther`

```

5331 \newcommand*{\GlsXtrStandaloneEntryOther}[2]{%
5332   \glstarget{#1}{\glossentrynameother{#1}{#2}}%
5333 }

```

`ntunsrtglossary` 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.

```

5334 \ifdef\@printgloss@checkexists
5335 {
5336   \newcommand*{\printunsrtglossary}{%
5337     \let\@printgloss@checkexists\@printgloss@checkexists@allowignored
5338     \@ifstar\s@printunsrtglossary\@printunsrtglossary
5339   }
5340 }
5341 {
5342   \newcommand*{\printunsrtglossary}{%
5343     \@ifstar\s@printunsrtglossary\@printunsrtglossary
5344   }
5345 }

```

ntunsrtglossary Unstarred version.

```
5346 \newcommand*{\@printunsrtglossary}[1][]{%  
5347   \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%  
5348 }
```

ntunsrtglossary Starred version.

```
5349 \newcommand*{\s@printunsrtglossary}[2][]{%  
5350   \begin{group}  
5351     #2%  
5352     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%  
5353   \end{group}  
5354 }
```

unsrtglossaries Similar to \printnoidxglossaries but it displays all entries defined for the given glossary without sorting.

```
5355 \newcommand*{\printunsrtglossaries}{%  
5356   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%  
5357 }
```

@unsrt@glossary

```
5358 \newcommand*{\@print@unsrt@glossary}{%  
5359   \glossarysection[\glossarytoctitle]{\glossarytitle}%  
5360   \glossarypreamble  
  
   check for empty list  
5361   \glxtrifemptyglossary{\@glo@type}%  
5362   {%  
5363     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%  
5364   }%  
5365   {%  
  
5366     \key@ifundefined{glossentry}{group}%  
5367     {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%  
5368     {\let\@gls@getgrouptitle\@glxtr@unsrt@getgrouptitle}%  
5369     \def\@gls@currentlettergroup{}}%
```

A loop within the tabular-like styles can cause problems, so move the loop outside.

```
5370 \def\@glxtr@doglossary{%  
5371   \begin{theglossary}%  
5372   \glossaryheader  
5373   \glsresetentrylist  
5374 }%  
5375 \expandafter\@for\expandafter\@glscurrententrylabel\expandafter  
5376   :\expandafter=\csname glist@\@glo@type\endcsname\do{%  
5377   \ifdefempty{\@glscurrententrylabel}  
5378   }%  
5379   {%
```

Provide a hook (for example to measure width).

```
5380   \let\@glxtr@process\@firstofone
```



```

5381      \let\printunsrtglossaryskipentry
5382      \@glxtr@printunsrtglossaryskipentry
5383      \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

5384      \glxtr@process
5385      {%
5386      \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.)

```

5387      \ifglshasparent{\glscurrententrylabel}{}%
5388      {%
5389      \@glxtr@checkgroup\glscurrententrylabel
5390      \expandafter\appto\expandafter\@glxtr@doglossary\expandafter
5391      {\@glxtr@groupheading}%
5392      }%
5393      \fi

5394      \protected@eappto\@glxtr@doglossary{%
5395      \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
5396      }%
5397      }%
5398      }%
5399      \appto\@glxtr@doglossary{\end{theglossary}}%
5400      \printunsrtglossarypredoglossary
5401      \@glxtr@doglossary
5402      }%
5403      \glossarypostamble
5404      }

```

`\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.

```

5405 \newcommand*{\printunsrtinnerglossary}[3] [] {%
5406   \begingroup
5407   \def\@glxtr@printglossopts{#1}%
5408   \def\@glo@type{\gldefaulttype}%
5409   \setkeys{printgloss}[title,toctitle,style,numberedsection,sort,label]{#1}%
5410   \let\currentglossary\@glo@type
5411   #2%
5412   \@print@unsrt@innerglossary
5413   #3%
5414   \endgroup
5415 }

```

srtglossarywrap

```

5416 \newenvironment{printunsrtglossarywrap}[1][]{%
5417 {%
5418   \def\@glstr@printglossopts{#1}%
5419   \def\@glo@type{\glstypedefault}%
5420   \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%
5421   \def\glossarytoctitle{\glossarytitle}%
5422   \let\org@glossarytitle\glossarytitle
5423   \def\@glossarystyle{%
5424     \ifx\@glossary@default@style\relax
5425       \GlossariesWarning{No default glossary style provided \MessageBreak
5426         for the glossary '\@glo@type'. \MessageBreak
5427         Using deprecated fallback. \MessageBreak
5428         To fix this set the style with \MessageBreak
5429         \string\setglossarystyle\space or use the \MessageBreak
5430         style key=value option}%
5431     \fi
5432   }%
5433   \def\gls@dotoc@title{\glstoc@title{\@glo@type}}%
5434   \let\org@glossaryentrynumbers\glossaryentrynumbers
5435   \@printgloss@setsort
5436   \setkeys{printgloss}{#1}%

```

The type key simply allows the title to be set if the title key isn't supplied.

```

5437 \ifglossaryexists*\@glo@type}%
5438 {%
5439   \ifx\glossarytitle\org@glossarytitle
5440   \else
5441     \expandafter\let\csname @glo@type\@glo@type @title\endcsname
5442       \glossarytitle
5443   \fi
5444   \let\currentglossary\@glo@type
5445 }%
5446 {}%
5447 \let\org@glossaryentrynumbers\glossaryentrynumbers
5448 \let\glsnonextpages\@glsnonextpages
5449 \let\glsnextpages\@glsnextpages
5450 \let\nopostdesc\@nopostdesc
5451 \gls@dotoc@title
5452 \@glossarystyle
5453 \let\gls@org@glossaryentryfield\glossentry
5454 \let\gls@org@glossarysubentryfield\subglossentry
5455 \renewcommand{\glossentry}[1]{%
5456   \protected@xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
5457   \gls@org@glossaryentryfield{##1}%
5458 }%
5459 \renewcommand{\subglossentry}[2]{%
5460   \protected@xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
5461   \gls@org@glossarysubentryfield{##1}{##2}%

```

```

5462 }%
5463 \@gls@preglossaryhook
5464 \glossarysection[\glossarytoctitle]{\glossarytitle}%
5465 \glossarypreamble
5466 \begin{theglossary}%
5467 \glossaryheader
5468 \glsresetentrylist
5469 }%
5470 {%
5471 \end{theglossary}%
5472 \glossarypostamble
5473 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
5474 \global\let\warn@noprintglossary\relax
5475 }

```

t@innerglossary This is much like \@print@unsrt@innerglossary but only contains what would normally be the content of the theglossary.

```

5476 \newcommand*{\@print@unsrt@innerglossary}{%

```

No section header or preamble.

```

5477 \glstrifemptyglossary{\@glo@type}%
5478 {%
5479 \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
5480 }%
5481 {%
5482 \key@ifundefined{glossentry}{group}%
5483 {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
5484 {\let\@gls@getgrouptitle\@glstr@unsrt@getgrouptitle}%
5485 \def\@gls@currentlettergroup{}}%

```

No header or reset.

```

5486 \def\@glstr@doglossary{%
5487 \expandafter\@for\expandafter\glscurrententrylabel\expandafter
5488 : \expandafter=\csname glolist@\@glo@type\endcsname\do{%
5489 \ifdefempty{\glscurrententrylabel}
5490 }%
5491 }%

```

Provide a hook (for example to measure width).

```

5492 \let\glstr@process\@firstofone
5493 \let\printunsrtglossaryskipentry
5494 \@glstr@printunsrtglossaryskipentry
5495 \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

5496 \glstr@process
5497 {%
5498 \ifglstr@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.)

```
5499      \ifglshasparent{\glscurrententrylabel}{}%
5500      {%
5501      \@glxtr@checkgroup\glscurrententrylabel
5502      \expandafter\appto\expandafter\@glxtr@doglossary\expandafter
5503      {\@glxtr@groupheading}%
5504      }%
5505      \fi

5506      \protected@eappto\@glxtr@doglossary{%
5507      \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
5508      }%
5509      }%
5510      }%
5511      \printunsrtglossarypredoglossary
5512      \@glxtr@doglossary
5513      }%
```

No postamble.

```
5514 }
```

entryprocesshook

```
5515 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{}
```

glossaryskipentry

```
5516 \newcommand*{\printunsrtglossaryskipentry}{%
5517   \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
5518   can only be used within \string\printunsrtglossaryentryprocesshook}{}%
5519 }
```

entryprocesshook

```
5520 \newcommand*{\@glxtr@printunsrtglossaryskipentry}{%
5521   \let\glxtr@process\@gobble
5522 }
```

rypredoglossary

```
5523 \newcommand*{\printunsrtglossarypredoglossary}{}%
```

lossary@handler

```
5524 \newcommand{\@printunsrt@glossary@handler}[1]{%
5525   \protected@xdef\glscurrententrylabel{#1}%
5526   \printunsrtglossaryhandler\glscurrententrylabel
5527 }
```

glossaryhandler

```
5528 \newcommand{\printunsrtglossaryhandler}[1]{%
5529   \glxtrunsrtdo{#1}%
5530 }
```

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 `\@gls@ifinlist` which ensures the label and list are fully expanded.

```
5531 \newrobustcmd*{\glxstriflabelinlist}[4]{%
5532   \protected@edef\@glxstr@doiflabelinlist{\noexpand\@gls@ifinlist{#1}{#2}}%
5533   \@glxstr@doiflabelinlist{#3}{#4}%
5534 }
```

srtglossaryunit

```
5535 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
5536   \s@printunsrtglossary[type=\glstypedefaulttype,#1]{%
5537     \printunsrtglossaryunitsetup{#2}%
5538   }%
5539 }
```

ossaryunitsetup

```
5540 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
5541   \renewcommand{\printunsrtglossaryhandler}[1]{%
5542     \glxstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
5543     {\glxstrtrnsrtdo{##1}}%
5544   }%
5545 }
```

Only the target names should have the prefixes adjusted as `\gls` etc need the original `\glolinkprefix`. The `\@gobble` part discards `\glolinkprefix`.

```
5546 \ifcsundef{theH#1}%
5547 {%
5548   \renewcommand*{\@glxstrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
5549 }%
5550 {%
5551   \renewcommand*{\@glxstrhypernameprefix}{record.#1.\csuse{theH#1}.\@gobble}%
5552 }%
5553 \renewcommand*{\glossarysection}[2][ ]{%
5554   \appto\glossarypostamble{\glspare\medskip\glspare}%
5555 }
```

srtglossaryunit

```
5556 \newcommand{\print@noop@unsrtglossaryunit}[2][ ]{%
5557   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
5558     requires the record=only or record=alsoindex package option}{}%
5559 }
```

t@getgrouptitle

```
5560 \newrobustcmd*{\@glxstr@unsrt@getgrouptitle}[2]{%
```

```

5561 \protected@edef\@glxstr@titlelabel{glxstr@grouptitle@#1}%
5562 \@onelevel@sanitize\@glxstr@titlelabel
5563 \ifcsdef{\@glxstr@titlelabel}
5564 {\letcs{#2}{\@glxstr@titlelabel}}%
5565 {\def#2{#1}}%
5566 }

```

`\glxstrunsrtdo` Provide a user-level call to `\@glxstr@noidx@do` to make it easier to define a new handler.

```

5567 \newcommand{\glxstrunsrtdo}{\@glxstr@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).

```

5568 \newcommand*{\glxtrgroupfield}{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 `\@glxstr@doglossary` is being constructed rather than in the handler.

`glxstr@checkgroup` The argument is the entry's label. (This block of code was formerly in `\@glxstr@noidx@do`.) Now that this is no longer within a tabular environment, the global definitions aren't needed. The result is now stored in `\@glxstr@groupheading`, which will be empty if no heading is required.

```

5569 \newcommand*{\@glxstr@checkgroup}[1]{%
5570   \def\@glxstr@groupheading{}%
5571   \key@ifundefined{glossentry}{group}%
5572   {%
5573     \letcs{\@gls@sort}{glo@\glsdetoklabel{#1}@sort}%
5574     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
5575   }%
5576   {%
5577     \protected@edef\@glo@thislettergrp{%
5578       \csuse{glo@\glsdetoklabel{#1}@glxtrgroupfield}}%
5579     }%
5580     \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
5581     {}%
5582     {%
5583       \ifdefempty{\@gls@currentlettergroup}{}%
5584       {\def\@glxstr@groupheading{\glsgroupskip}}%
5585       \protected@eappto\@glxstr@groupheading{%
5586         \noexpand\glsgroupheading{\expandonce\@glo@thislettergrp}%
5587       }%
5588     }%
5589     \let\@gls@currentlettergroup\@glo@thislettergrp
5590 }

```

`trLocationField` Stores the internal name of the location field.

```

5591 \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.

```

5592 \newcommand{\@glsxtr@noidx@do}[1]{%
5593   \ifglstryexists{#1}%
5594   {%
5595     \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
5596     \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@\GlsXtrLocationField}%

```

Use level number to determine whether or not this entry has a parent.

```

5597     \gls@level=\numexpr\csuse{glo@\glsdetoklabel{#1}@level}+\@glsxtr@leveloffset\relax
5598     \ifnum\gls@level>0
5599       \let\@glsxtr@ifischild\@firstoftwo
5600     \else
5601       \let\@glsxtr@ifischild\@secondoftwo
5602     \fi

```

Some glossary styles (such as topicmccls) save the level using \def so make sure \gls@level is expanded before being passed to \subglossentry.

```

5603   \@glsxtr@ifischild
5604   {%
5605     \ifdefvoid{\@gls@location}%
5606     {%
5607       \ifdefvoid{\@gls@loclist}%
5608       {%
5609         \expandafter\subglossentry\expandafter{\number\gls@level}{#1}{}%
5610       }%
5611     }%
5612     \expandafter\subglossentry\expandafter{\number\gls@level}{#1}%
5613     {%
5614       \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5615     }%
5616   }%
5617 }%
5618 {%
5619   \expandafter\subglossentry\expandafter
5620   {\number\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
5621 }%
5622 }%
5623 {%
5624   \ifdefvoid{\@gls@location}%
5625   {%
5626     \ifdefvoid{\@gls@loclist}
5627     {%
5628       \glossentry{#1}{}%
5629     }%
5630   }%
5631   \glossentry{#1}%
5632   {%
5633     \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%

```

```

5634         }%
5635     }%
5636 }%
5637 {%
5638     \glossentry{#1}%
5639     {%
5640         \glossaryentrynumbers{\@gls@location}%
5641     }%
5642 }%
5643 }%
5644 }%
5645 {}%
5646 }

```

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.

`\glsxtrnewgls`

```

5647 \newcount\@glsxtrnewgls@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`

```

5648 \newcommand*{\@glsxtr@providenewgls}{%
5649     \protected@write\@auxout{}{\string\providecommand{\string\@glsxtr@newglslike}[2]{}}%
5650     \let\@glsxtr@providenewgls\relax
5651 }

```

`identifyglslike` Identify the command given in the second argument for the benefit of **bib2gls**.

```

5652 \newcommand{\glsxtridentifyglslike}[2]{%
5653     \ifdefequal\@glsxtr@record@setting\@glsxtr@record@setting@off
5654     }%
5655     {%
5656         \@glsxtr@providenewgls
5657         \protected@write\@auxout{}{\string\@glsxtr@newglslike{#1}{\string#2}}%
5658     }%
5659 }

```

`\@glsxtrnewgls`

`\glsxtrnewgls[<options>]{<prefix>}{<cs>}{<inner cs name>}`

```

5660 \newcommand*{\@glsxtrnewgls}[4]{%
5661     \ifdef{#3}%

```



```

5662 {%
5663   \PackageError{glossaries-extra}{Command \string#3\space already
5664 defined}{}%
5665 }%
5666 {%

  Write information to the aux file for bib2gls.

5667   \glstridentifyglslike{#2}{#3}%
5668   \ifcsdef{@#4like@#2}%
5669   {%
5670     \advance\@glstrnewgls@inner by \@ne
5671     \def\@glstrnewgls@innercsname{@#4like\number\@glstrnewgls@inner @#2}%
5672   }%
5673   {\def\@glstrnewgls@innercsname{@#4like@#2}}%
5674   \expandafter\newrobustcmd\expandafter*\expandafter
5675     #3\expandafter{\expandafter\@glshyp@opt\csname\@glstrnewgls@innercsname\endcsname}%
5676   \ifstrempy{#1}%
5677   {%
5678     \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2][]{%
5679       \new@ifnextchar[%
5680         {\csname @#4@\endcsname{##1}{#2##2}}%
5681         {\csname @#4@\endcsname{##1}{#2##2} []}%
5682       }%
5683     }%
5684   {%
5685     \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2][]{%
5686       \new@ifnextchar[%
5687         {\csname @#4@\endcsname{#1,##1}{#2##2}}%
5688         {\csname @#4@\endcsname{#1,##1}{#2##2} []}%
5689       }%
5690     }%
5691   }%
5692 }

```

\glstrnewgls

`\glstrnewgls[<options>]{<prefix>}{<cs>}`

The first argument prepends to the options and the second argument is the prefix.

```

5693 \newrobustcmd*{\glstrnewgls}[3][]{%
5694   \@glstrnewgls{#1}{#2}{#3}{gls}%
5695 }

```

lsxtrnewglslike 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.

```

5696 \newrobustcmd*{\glstrnewglslike}[6][]{%
5697   \@glstrnewgls{#1}{#2}{#3}{gls}%

```

```

5698 \@glxstrnewgls{#1}{#2}{#4}{glspl}%
5699 \@glxstrnewgls{#1}{#2}{#5}{Gls}%
5700 \@glxstrnewgls{#1}{#2}{#6}{Glspl}%
5701 }

```

lsxtrnewGLSlike 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.

```

5702 \newrobustcmd*{\glxtrnewGLSlike}[4][]{%
5703 \@glxstrnewgls{#1}{#2}{#3}{GLS}%
5704 \@glxstrnewgls{#1}{#2}{#4}{GLSpl}%
5705 }

```

\glxtrnewrgls As \glxtrnewgls but for \rgls.

```

5706 \newrobustcmd*{\glxtrnewrgls}[3][]{%
5707 \@glxstrnewgls{#1}{#2}{#3}{rgls}%
5708 }

```

sxtrnewrglslike As \glxtrnewrglslike but for \rgls etc.

```

5709 \newrobustcmd*{\glxtrnewrglslike}[6][]{%
5710 \@glxstrnewgls{#1}{#2}{#3}{rgls}%
5711 \@glxstrnewgls{#1}{#2}{#4}{rglspl}%
5712 \@glxstrnewgls{#1}{#2}{#5}{rGls}%
5713 \@glxstrnewgls{#1}{#2}{#6}{rGlspl}%
5714 }

```

sxtrnewrGLSlike As \glxtrnewGLSlike but for \rGLS etc.

```

5715 \newrobustcmd*{\glxtrnewrGLSlike}[4][]{%
5716 \@glxstrnewgls{#1}{#2}{#3}{rGLS}%
5717 \@glxstrnewgls{#1}{#2}{#4}{rGLSpl}%
5718 }

```

Provide easy access to record count fields.

totalRecordCount Access total record count. This is designed to be expandable. The argument is the label.

```

5719 \newcommand*{\GlsXtrTotalRecordCount}[1]{%
5720 \ifcsdef{glo@\glsdetoklabel{#1}@recordcount}%
5721 {\csname glo@\glsdetoklabel{#1}@recordcount\endcsname}%
5722 {0}%
5723 }

```

sXtrRecordCount Access record count for a particular counter. The first argument is the label. The second argument is the counter name.

```

5724 \newcommand*{\GlsXtrRecordCount}[2]{%
5725 \ifcsdef{glo@\glsdetoklabel{#1}@recordcount.#2}%
5726 {\csname glo@\glsdetoklabel{#1}@recordcount.#2\endcsname}%
5727 {0}%
5728 }

```

`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.

```
5729 \newcommand*{\GlsXtrLocationRecordCount}[3]{%
5730   \ifcsdef{glo@\glsetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}}{%
5731     {\csname glo@\glsetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}\endcsname}%
5732     {0}}%
5733 }
```

`trdetoklocation`

```
5734 \newcommand*{\glxtrdetoklocation}[1]{#1}
```

`ablerecordcount`

```
5735 \newcommand*{\glxtrenablerecordcount}{%
5736   \renewcommand*{\gls}{\rgls}%
5737   \renewcommand*{\Gls}{\rGls}%
5738   \renewcommand*{\glspl}{\rglspl}%
5739   \renewcommand*{\Glspl}{\rGlspl}%
5740   \renewcommand*{\GLS}{\rGLS}%
5741   \renewcommand*{\GLSpl}{\rGLSpl}%
5742 }
```

`ordtriggervalue` The value used by the record trigger test. The argument is the entry's label.

```
5743 \newcommand*{\glxtrrecordtriggervalue}[1]{%
5744   \GlsXtrTotalRecordCount{#1}%
5745 }
```

`dCountAttribute`

```
5746 \newcommand*{\GlsXtrSetRecordCountAttribute}[2]{%
5747   \@for\@glxtr@cat:=#1\do
5748   {%
5749     \ifdefempty{\@glxtr@cat}{}%
5750     {%
5751       \glsssetcategoryattribute{\@glxtr@cat}{recordcount}{#2}%
5752     }%
5753   }%
5754 }
```

`ifrecordtrigger`

`\glxtrifrecordtrigger{<label>}{<trigger format>}{<normal>}`

```
5755 \newcommand*{\glxtrifrecordtrigger}[3]{%
5756   \glshasattribute{#1}{recordcount}%
5757   {%
```

```

5758 \ifnum\glstrrecordtriggervalue{#1}>\glsggetattribute{#1}{recordcount}\relax
5759 #3%
5760 \else
5761 #2%
5762 \fi
5763 }%
5764 {#3}%
5765 }

```

strigger@record Still need a record to ensure that bib2gls selects the entry.

```

5766 \newcommand*{\@glstr@rglstrigger@record}[3]{%
5767 \protected@edef\glslabel{\glsdetoklabel{#2}}%
5768 \let\@glsl@link@label\glslabel
5769 \def\@glstr@thevalue{%
5770 \def\@glstr@theHvalue{\@glstr@thevalue}%
5771 \def\@glslnumberformat{glstriggerrecordformat}%
5772 \protected@edef\@glsl@counter{\csname glo@\glslabel @counter\endcsname}%
5773 \protected@edef\glstype{\csname glo@\glslabel @type\endcsname}%
5774 \def\@glstr@thevalue{%
5775 \def\@glstr@theHvalue{\@glstr@thevalue}%
5776 \glstrinitwrgloss
5777 \glslinkpresetkeys
5778 \setkeys{glslink}{#1}%
5779 \glslinkpostsetkeys
5780 \ifdefempty{\@glstr@thevalue}%
5781 {%
5782 \@glsl@saveentrycounter
5783 }%
5784 {%
5785 \let\theglslentrycounter\@glstr@thevalue
5786 \def\theHglslentrycounter{\@glstr@theHvalue}%
5787 }%
5788 \ifglstrinitwrglossbefore
5789 \@do@wrglossary{#2}%
5790 \fi
5791 #3%
5792 \ifglstrinitwrglossbefore
5793 \else
5794 \@do@wrglossary{#2}%
5795 \fi
5796 \ifKV@glslink@local
5797 \glsllocalunset{#2}%
5798 \else
5799 \glslunset{#2}%
5800 \fi
5801 }

```

gerrecordformat Typically won't be used as it should be recognised as a special type of ignored location by bib2gls.

```

5802 \newcommand*{\glstriggerrecordformat}[1]{%

\rgls
5803 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}

\@rgls
5804 \newcommand*{\@rgls}[2][{}]{%
5805   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}]%
5806 }

\@rgls@
5807 \def\@rgls@#1#2[#3]{%
5808   \glstrifrecordtrigger{#2}%
5809   {%
5810     \@glstr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
5811   }%
5812   {%
5813     \@gls@{#1}{#2}[#3]%
5814   }%
5815 }%

\rglspl
5816 \newrobustcmd*{\rglspl}{\@gls@hyp@opt\@rglspl}

\@rglspl
5817 \newcommand*{\@rglspl}[2][{}]{%
5818   \new@ifnextchar[{\@rglspl@{#1}{#2}}{\@rglspl@{#1}{#2}[]}]%
5819 }

\@rglspl@
5820 \def\@rglspl@#1#2[#3]{%
5821   \glstrifrecordtrigger{#2}%
5822   {%
5823     \@glstr@rglstrigger@record{#1}{#2}{\rglsplformat{#2}{#3}}%
5824   }%
5825   {%
5826     \@glspl@{#1}{#2}[#3]%
5827   }%
5828 }%

\rgls
5829 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}

\@rgls
5830 \newcommand*{\@rgls}[2][{}]{%
5831   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}]%
5832 }

```

```

\@rGls@
5833 \def\@rGls@#1#2[#3]{%
5834   \glxtrifrecordtrigger{#2}%
5835   {%
5836     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsformat{#2}{#3}}%
5837   }%
5838   {%
5839     \@Gls@{#1}{#2}[#3]%
5840   }%
5841 }%

\rGlspl
5842 \newrobustcmd*{\rGlspl}{\@glshyp@opt\rGlspl}

\@rGlspl
5843 \newcommand*{\@rGlspl}[2][{}]{%
5844   \new@ifnextchar[{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2}[]}%
5845 }

\@rGlspl@
5846 \def\@rGlspl@#1#2[#3]{%
5847   \glxtrifrecordtrigger{#2}%
5848   {%
5849     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
5850   }%
5851   {%
5852     \@Glspl@{#1}{#2}[#3]%
5853   }%
5854 }%

\rGLS
5855 \newrobustcmd*{\rGLS}{\@glshyp@opt\rGLS}

\@rGLS
5856 \newcommand*{\@rGLS}[2][{}]{%
5857   \new@ifnextchar[{\@rGLS@{#1}{#2}}{\@rGLS@{#1}{#2}[]}%
5858 }

\@rGLS@
5859 \def\@rGLS@#1#2[#3]{%
5860   \glxtrifrecordtrigger{#2}%
5861   {%
5862     \@glxtr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%
5863   }%
5864   {%
5865     \@GLS@{#1}{#2}[#3]%
5866   }%
5867 }%

```

```

\rglSpl
5868 \newrobustcmd*{\rglSpl}{\@gls@hyp@opt\@rglSpl}

\@rglSpl
5869 \newcommand*{\@rglSpl}[2][{}]{%
5870   \new@ifnextchar[{\@rglSpl@{#1}{#2}}{\@rglSpl@{#1}{#2}[]}%
5871 }

\@rglSpl@
5872 \def\@rglSpl@#1#2[#3]{%
5873   \glstriferecordtrigger{#2}%
5874   {%
5875     \@glstr@rglstrigger@record{#1}{#2}{\rglSplformat{#2}{#3}}%
5876   }%
5877   {%
5878     \@rglSpl@{#1}{#2}[#3]%
5879   }%
5880 }%

\rglsformat
5881 \newcommand*{\rglsformat}[2]{%
5882   \glsifregular{#1}
5883   {\glsentryfirst{#1}}%
5884   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
5885 }

\rglsplformat
5886 \newcommand*{\rglsplformat}[2]{%
5887   \glsifregular{#1}
5888   {\glsentryfirstplural{#1}}%
5889   {\ifglshaslong{#1}{\glsentrylongplural{#1}}{\glsentryfirstplural{#1}}}%#2%
5890 }

\rglsformat
5891 \newcommand*{\rglsformat}[2]{%
5892   \glsifregular{#1}
5893   {\Glsentryfirst{#1}}%
5894   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
5895 }

\rglsplformat
5896 \newcommand*{\rglsplformat}[2]{%
5897   \glsifregular{#1}
5898   {\Glsentryfirstplural{#1}}%
5899   {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}}%#2%
5900 }

```

`\rGLSformat`

```
5901 \newcommand*\rGLSformat}[2]{%
5902   \expandafter\mfirstucMakeUppercase\expandafter{\rGLSformat{#1}{#2}}%
5903 }
```

`\rGLSplformat`

```
5904 \newcommand*\rGLSplformat}[2]{%
5905   \expandafter\mfirstucMakeUppercase\expandafter{\rGLSplformat{#1}{#2}}%
5906 }
```

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 `\glsxtr@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`.

```
5907 \newcommand*\@glsxtr@do@inc@linkcount{%
```

Does this entry have the linkcount attribute set?

```
5908   \glsifattribute{\glslabel}{linkcount}{true}%
```

```
5909   {%
```

Does the counter exist?

```
5910     \ifcsdef{c@glxtr@linkcount@\glslabel}{}%
```

```
5911     {%
```

Counter doesn’t exist, so define it.

```
5912       \newcounter{glxtr@linkcount@\glslabel}%
```

If linkcountmaster is set, add to counter reset.

```
5913       \glshasattribute{\glslabel}{linkcountmaster}%
```

```
5914       {%
```

Need to ensure values are fully expanded.

```
5915         \begingroup
```

```
5916         \edef\@glo@tmp{\endgroup\noexpand\@addtoreset{glxtr@linkcount@\glslabel}%
```

```
5917         {\glsgetattribute{\glslabel}{linkcountmaster}}}%
```

```
5918         \@glo@tmp
```

```
5919       }%
```

```
5920       {}%
```

```
5921     }%
```

Increment counter:

```
5922   \glsxtrinclinkcounter{glxtr@linkcount@\glslabel}%
```



```

5923 }%
5924 {}%
5925 }

```

`\incLinkCounter` May be redefined to use `\refstepcounter` if required.

```

5926 \newcommand*\GlsXtrIncLinkCounter}[1]{\stepcounter{#1}}

```

`\linkCounterValue` Expands to the associated link counter register or 0 if not defined.

```

5927 \newcommand*\GlsXtrLinkCounterValue}[1]{%
5928   \ifcsundef{c@glsextr@linkcount@#1}{0}{\csname c@glsextr@linkcount@#1\endcsname}%
5929 }

```

`\TheLinkCounter` Expands to the display value of the associated link counter or 0 if not defined.

```

5930 \newcommand*\GlsXtrTheLinkCounter}[1]{%
5931   \ifcsundef{theglsxtr@linkcount@#1}{0}%
5932   {\csname theglsextr@linkcount@#1\endcsname}%
5933 }

```

`\ifLinkCounterDef` Tests if the counter has been defined

```

5934 \newcommand*\GlsXtrIfLinkCounterDef}[3]{%
5935   \ifcsundef{theglsxtr@linkcount@#1}{#3}{#2}%
5936 }

```

`\LinkCounterName` Expands to the associated link counter name. (No check for existence.)

```

5937 \newcommand*\GlsXtrLinkCounterName}[1]{glsextr@linkcount@#1}

```

`\enableLinkCounting`

`\GlsXtrEnableLinkCounting[<master counter>]{<categories>}`

Enable link counting for the given categories.

```

5938 \newcommand*\GlsXtrEnableLinkCounting}[2][ ]{%
5939   \let\glsextr@inc@linkcount\@glsextr@do@inc@linkcount
5940   \@for\@glsextr@label:=#2\do
5941   {%
5942     \glsssetcategoryattribute{\@glsextr@label}{linkcount}{true}%
5943     \ifstrempy{#1}{}%
5944     {%
5945       \ifcsundef{c@#1}%
5946       {\@nocounterr{#1}}%
5947       {\glsssetcategoryattribute{\@glsextr@label}{linkcountmaster}{#1}}%
5948     }%
5949   }%
5950 }
5951 \@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.

```
5952 \@ifpackageloaded{glossaries-accsupp}
5953 {
```

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

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

```
5954 \newcommand*{\glsaccessname}[1]{%
5955 \glsnameaccessdisplay
5956 {%
5957 \glstryname{#1}%
5958 }%
5959 {#1}%
5960 }
```

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

```
5961 \newcommand*{\Glsaccessname}[1]{%
5962 \glsnameaccessdisplay
5963 {%
5964 \Glsstryname{#1}%
5965 }%
5966 {#1}%
5967 }
```

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

```
5968 \newcommand*{\GLSaccessname}[1]{%
5969 \glsnameaccessdisplay
5970 {%
5971 \mfirstucMakeUppercase{\glstryname{#1}}%
5972 }%
5973 {#1}%
5974 }
```

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

```
5975 \newcommand*{\glsaccesstext}[1]{%
5976 \glstextaccessdisplay
5977 {%
5978 \glstrytext{#1}%
5979 }%
5980 {#1}%
5981 }
```

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

```
5982 \newcommand*{\Glsaccesstext}[1]{%
5983   \glstextaccessdisplay
5984   {%
5985     \Glsentrytext{#1}%
5986   }%
5987   {#1}%
5988 }
```

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

```
5989 \newcommand*{\GLSaccesstext}[1]{%
5990   \glstextaccessdisplay
5991   {%
5992     \mfirstucMakeUppercase{\Glsentrytext{#1}}%
5993   }%
5994   {#1}%
5995 }
```

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

```
5996 \newcommand*{\glsaccessplural}[1]{%
5997   \glspluralaccessdisplay
5998   {%
5999     \Glsentryplural{#1}%
6000   }%
6001   {#1}%
6002 }
```

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

```
6003 \newcommand*{\GLSaccessplural}[1]{%
6004   \glspluralaccessdisplay
6005   {%
6006     \Glsentryplural{#1}%
6007   }%
6008   {#1}%
6009 }
```

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

```
6010 \newcommand*{\GLSaccessplural}[1]{%
6011   \glspluralaccessdisplay
6012   {%
6013     \mfirstucMakeUppercase{\Glsentryplural{#1}}%
6014   }%
6015   {#1}%
6016 }
```

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

```

6017 \newcommand*{\glsaccessfirst}[1]{%
6018   \glsfirstaccessdisplay
6019   {%
6020     \glstryfirst{#1}%
6021   }%
6022   {#1}%
6023 }

```

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

```

6024 \newcommand*{\Glsaccessfirst}[1]{%
6025   \glsfirstaccessdisplay
6026   {%
6027     \Glsstryfirst{#1}%
6028   }%
6029   {#1}%
6030 }

```

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

```

6031 \newcommand*{\GLSaccessfirst}[1]{%
6032   \glsfirstaccessdisplay
6033   {%
6034     \mfirstucMakeUppercase{\glstryfirst{#1}}%
6035   }%
6036   {#1}%
6037 }

```

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

```

6038 \newcommand*{\glsaccessfirstplural}[1]{%
6039   \glsfirstpluralaccessdisplay
6040   {%
6041     \glstryfirstplural{#1}%
6042   }%
6043   {#1}%
6044 }

```

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

```

6045 \newcommand*{\Glsaccessfirstplural}[1]{%
6046   \glsfirstpluralaccessdisplay
6047   {%
6048     \Glsstryfirstplural{#1}%
6049   }%
6050   {#1}%
6051 }

```

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

```

6052 \newcommand*{\GLSaccessfirstplural}[1]{%

```

```

6053   \glsfirstpluralaccessdisplay
6054   {%
6055       \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
6056   }%
6057   {#1}%
6058   }

```

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

```

6059   \newcommand*{\glsaccesssymbol}[1]{%
6060       \glssymbolaccessdisplay
6061       {%
6062           \glsentrysymbol{#1}%
6063       }%
6064       {#1}%
6065   }

```

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

```

6066   \newcommand*{\Glsaccesssymbol}[1]{%
6067       \glssymbolaccessdisplay
6068       {%
6069           \Glsentrysymbol{#1}%
6070       }%
6071       {#1}%
6072   }

```

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

```

6073   \newcommand*{\GLSaccesssymbol}[1]{%
6074       \glssymbolaccessdisplay
6075       {%
6076           \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
6077       }%
6078       {#1}%
6079   }

```

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

```

6080   \newcommand*{\glsaccesssymbolplural}[1]{%
6081       \glssymbolpluralaccessdisplay
6082       {%
6083           \glsentrysymbolplural{#1}%
6084       }%
6085       {#1}%
6086   }

```

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

```

6087   \newcommand*{\Glsaccesssymbolplural}[1]{%
6088       \glssymbolpluralaccessdisplay

```

```

6089   {%
6090     \Glsentrysymbolplural{#1}%
6091   }%
6092   {#1}%
6093 }

```

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

```

6094 \newcommand*{\GLSaccesssymbolplural}[1]{%
6095   \glssymbolpluralaccessdisplay
6096   {%
6097     \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
6098   }%
6099   {#1}%
6100 }

```

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

```

6101 \newcommand*{\glsaccessdesc}[1]{%
6102   \glsdescriptionaccessdisplay
6103   {%
6104     \glentrydesc{#1}%
6105   }%
6106   {#1}%
6107 }

```

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

```

6108 \newcommand*{\Glsaccessdesc}[1]{%
6109   \glsdescriptionaccessdisplay
6110   {%
6111     \Glsentrydesc{#1}%
6112   }%
6113   {#1}%
6114 }

```

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

```

6115 \newcommand*{\GLSaccessdesc}[1]{%
6116   \glsdescriptionaccessdisplay
6117   {%
6118     \mfirstucMakeUppercase{\glentrydesc{#1}}%
6119   }%
6120   {#1}%
6121 }

```

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

```

6122 \newcommand*{\glsaccessdescplural}[1]{%
6123   \glsdescriptionpluralaccessdisplay
6124   {%
6125     \glentrydescplural{#1}%

```

```

6126     }%
6127     {#1}%
6128 }

```

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

```

6129 \newcommand*\Glsaccessdescplural[1]{%
6130   \glsdescriptionpluralaccessdisplay
6131   {%
6132     \Glsentrydescplural{#1}%
6133   }%
6134   {#1}%
6135 }

```

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

```

6136 \newcommand*\GLSaccessdescplural[1]{%
6137   \glsdescriptionpluralaccessdisplay
6138   {%
6139     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
6140   }%
6141   {#1}%
6142 }

```

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

```

6143 \newcommand*\glsaccessshort[1]{%
6144   \glsshortaccessdisplay
6145   {%
6146     \glsentryshort{#1}%
6147   }%
6148   {#1}%
6149 }

```

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

```

6150 \newcommand*\Glsaccessshort[1]{%
6151   \glsshortaccessdisplay
6152   {%
6153     \Glsentryshort{#1}%
6154   }%
6155   {#1}%
6156 }

```

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

```

6157 \newcommand*\GLSAccessshort[1]{%
6158   \glsshortaccessdisplay
6159   {%
6160     \mfirstucMakeUppercase{\glsentryshort{#1}}%
6161   }%

```

```

6162     {#1}%
6163 }

```

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

```

6164 \newcommand*{\lsaccessshortpl}[1]{%
6165     \glshortpluralaccessdisplay
6166     {%
6167         \glentryshortpl{#1}%
6168     }%
6169     {#1}%
6170 }

```

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

```

6171 \newcommand*{\lSaccessshortpl}[1]{%
6172     \glshortpluralaccessdisplay
6173     {%
6174         \Glentryshortpl{#1}%
6175     }%
6176     {#1}%
6177 }

```

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

```

6178 \newcommand*{\LSaccessshortpl}[1]{%
6179     \glshortpluralaccessdisplay
6180     {%
6181         \mfirstucMakeUppercase{\glentryshortpl{#1}}%
6182     }%
6183     {#1}%
6184 }

```

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

```

6185 \newcommand*{\glaccesslong}[1]{%
6186     \glslongaccessdisplay{\glentrylong{#1}}{#1}%
6187 }

```

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

```

6188
6189 \newcommand*{\Glaccesslong}[1]{%
6190     \glslongaccessdisplay{\Glentrylong{#1}}{#1}%
6191 }

```

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

```

6192 \newcommand*{\GLSaccesslong}[1]{%
6193     \glslongaccessdisplay
6194     {%
6195         \mfirstucMakeUppercase{\glentrylong{#1}}%
6196     }%

```



```

6197     {#1}%
6198 }

```

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

```

6199 \newcommand*{\glsaccesslongpl}[1]{%
6200   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
6201 }

```

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

```

6202
6203 \newcommand*{\GLsaccesslongpl}[1]{%
6204   \glslongpluralaccessdisplay{\GLsentrylongpl{#1}}{#1}%
6205 }

```

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

```

6206 \newcommand*{\GLSaccesslongpl}[1]{%
6207   \glslongpluralaccessdisplay
6208   {%
6209     \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
6210   }%
6211   {#1}%
6212 }

```

Keys for accessibility support.

```

6213 \define@key{glsxtrabbrv}{access}{%
6214   \def\@gls@nameaccess{#1}%
6215 }
6216 \define@key{glsxtrabbrv}{textaccess}{%
6217   \def\@gls@textaccess{#1}%
6218 }
6219 \define@key{glsxtrabbrv}{pluralaccess}{%
6220   \def\@gls@pluralaccess{#1}%
6221 }
6222 \define@key{glsxtrabbrv}{firstaccess}{%
6223   \def\@gls@firstaccess{#1}%
6224 }
6225 \define@key{glsxtrabbrv}{firstpluralaccess}{%
6226   \def\@gls@firstpluralaccess{#1}%
6227 }
6228 \define@key{glsxtrabbrv}{shortaccess}{%
6229   \def\@gls@shortaccess{#1}%
6230 }
6231 \define@key{glsxtrabbrv}{shortpluralaccess}{%
6232   \def\@gls@shortaccesspl{#1}%
6233 }

```

```

6234 \define@key{glsxtrabrv}{longaccess}{%
6235   \def\@gls@longaccess{#1}%
6236 }
6237 \define@key{glsxtrabrv}{shortlonglaccess}{%
6238   \def\@gls@longaccesspl{#1}%
6239 }

```

@initaccesskeys

```

6240 \newcommand*{\@gls@initaccesskeys}{%
6241   \def\@gls@nameaccess{}%
6242   \def\@gls@textaccess{}%
6243   \def\@gls@pluralaccess{}%
6244   \def\@gls@firstaccess{}%
6245   \def\@gls@firstpluralaccess{}%
6246   \def\@gls@shortaccess{}%
6247   \def\@gls@shortaccesspl{}%
6248   \def\@gls@longaccess{}%
6249   \def\@gls@longaccesspl{}%
6250 }

```

ssattribute@set

```
\gls@ifaccessattribute@set{<attribute>}{<true>}{<false>}
```

```

6251 \newcommand*{\@gls@ifaccessattribute@set}[3]{%
6252   \glsifcategoryattribute{\glscategorylabel}{access#1}{true}%
6253   {#2}%
6254   {%
6255     \glsifcategoryattribute{\glscategorylabel}{access#1}{false}%
6256     {#3}%
6257     {%
6258       \glsifcategoryattribute{\glscategorylabel}{#1}{true}%
6259       {#2}%
6260       {#3}%
6261     }%
6262   }%
6263 }

```

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.

```
6264 \def\glsdefaultshortaccess#1#2{#1 (#2)}
```

signactualsetup

```
6265 \newcommand{\glsxtrassignactualsetup}{%
6266 \let\@empty
6267 \let\emph\@firstofone
6268 \let\textbf\@firstofone
6269 \let\textmd\@firstofone
6270 \let\textit\@firstofone
6271 \let\textsl\@firstofone
6272 \let\textsc\@firstofone
6273 \let\textrm\@firstofone
6274 \let\textsf\@firstofone
6275 \let\texttt\@firstofone
6276 }
```

s@assign@actual

```
6277 \ifdef\pdfstringdef
6278 {
6279 \newcommand{\@gls@assign@actual}{%
6280 \begingroup
6281 \glsxtrassignactualsetup
6282 \pdfstringdef\@gls@actualshort{\glsxtrorgshort}%
6283 \pdfstringdef\@gls@actuaallong{\glsxtrorglong}%
6284 \pdfstringdef\@gls@actualshortpl{\@gls@shortpl}%
6285 \pdfstringdef\@gls@actuaallongpl{\@gls@longpl}%
6286 \protected@edef\@gls@tmp{\endgroup
6287 \def\noexpand\@gls@actualshort{\expandonce\@gls@actualshort}%
6288 \def\noexpand\@gls@actuaallong{\expandonce\@gls@actuaallong}%
6289 \def\noexpand\@gls@actualshortpl{\expandonce\@gls@actualshortpl}%
6290 \def\noexpand\@gls@actuaallongpl{\expandonce\@gls@actuaallongpl}%
6291 }%
6292 \@gls@tmp
6293 }
6294 }
6295 {
6296 \newcommand{\@gls@assign@actual}{%
6297 \begingroup
6298 \glsxtrassignactualsetup
6299 \protected@edef\@gls@tmp{\endgroup
6300 \def\noexpand\@gls@actualshort{\glsxtrorgshort}%
6301 \def\noexpand\@gls@actuaallong{\glsxtrorglong}%
6302 \def\noexpand\@gls@actualshortpl{\@gls@shortpl}%
6303 \def\noexpand\@gls@actuaallongpl{\@gls@longpl}%
6304 }%
6305 \@gls@tmp
```

```

6306     }
6307 }

\let\short@access  Renamed \@gls@setup@default@access and removed argument since it can be obtained
                    from \glxstrorgshort.

\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.

6308   \newcommand{\@gls@setup@default@access}{%
6309     \@gls@assign@actual
6310     \ifdefempty\@gls@shortaccess
6311     {%
        Check if the accessinsertdots attribute has been set but only if shortaccess hasn't been set.
6312         \@gls@ifaccessattribute@set{insertdots}%
6313         {%
6314             \expandafter\@glxtr@insertdots\expandafter\@gls@actualshort\expandafter
6315             {\@gls@actualshort}%
6316         }%
6317         {}%
6318         \ifdefempty\@gls@longaccess
6319         {%
6320             \protected@edef\@gls@shortaccess{\glsdefaultshortaccess
6321             {\expandonce\@gls@actuallong}{\expandonce\@gls@actualshort}}%
6322         }%
6323         {%
6324             \protected@edef\@gls@shortaccess{\glsdefaultshortaccess
6325             {\expandonce\@gls@longaccess}{\expandonce\@gls@actualshort}}%
6326         }%
6327         \eappto\ExtraCustomAbbreviationFields{shortaccess={\@gls@shortaccess},}%

        If shortaccessplural hasn't been set, assign plural form.
6328         \ifdefempty\@gls@shortaccesspl
6329         {%
6330             \@gls@ifaccessattribute@set{aposplural}%
6331             {%
6332                 \expandafter\def\expandafter\@gls@shortaccesspl\expandafter{%
6333                     \@gls@actualshort'\glxstrabbrvpluralsuffix}%
6334             }%
6335             {%
6336                 \@gls@ifaccessattribute@set{noshortplural}%
6337                 {%
6338                     \let\@gls@shortaccesspl\@gls@shortaccess
6339                 }%
6340                 {%
6341                     \let\@gls@shortaccesspl\@gls@actualshortpl
6342                 }%
6343             }%
6344             \ifdefempty\@gls@longaccesspl
6345             {%

```

```

6346     \protected@edef\@gls@shortaccesspl{\glsdefaultshortaccess
6347         {\expandonce\@gls@actuallongpl}{\expandonce\@gls@actualshortpl}}%
6348     }%
6349     {%
6350     \protected@edef\@gls@shortaccesspl{\glsdefaultshortaccess
6351         {\expandonce\@gls@longaccesspl}{\expandonce\@gls@actualshort}}%
6352     }%
6353     \eappto\ExtraCustomAbbreviationFields{shortpluralaccess={\@gls@shortaccesspl},}%
6354     }%
6355     {}%
6356     }%
6357     {%
6358     \ifdefempty\@gls@shortaccesspl
6359     {\let\@gls@shortaccesspl\@gls@shortaccess}%
6360     }%
6361     }%

```

If access key hasn't been set, check if the nameshortaccess attribute has been set.

```

6362     \ifdefempty\@gls@nameaccess
6363     {%
6364     \glsifcategoryattribute{\glscategorylabel}{nameshortaccess}{true}%
6365     {%
6366     \eappto\ExtraCustomAbbreviationFields{access={\@gls@shortaccess},}%
6367     }%
6368     {}%
6369     }%
6370     {}%

```

If textaccess key hasn't been set, check if the textshortaccess attribute has been set.

```

6371     \ifdefempty\@gls@textaccess
6372     {%
6373     \glsifcategoryattribute{\glscategorylabel}{textshortaccess}{true}%
6374     {%
6375     \eappto\ExtraCustomAbbreviationFields{textaccess={\@gls@shortaccess},}%
6376     }%
6377     {}%
6378     }%
6379     {}%
6380     \ifdefempty\@gls@pluralaccess
6381     {%
6382     \glsifcategoryattribute{\glscategorylabel}{textshortaccess}{true}%
6383     {%
6384     \eappto\ExtraCustomAbbreviationFields{%
6385         pluralaccess={\@gls@shortaccesspl},%
6386     }%
6387     }%
6388     {}%
6389     }%
6390     {}%

```

If firstaccess key hasn't been set, check if the firstshortaccess attribute has been set.

```

6391 \ifdefempty\@gls@firstaccess
6392 {%
6393 \glsifcategoryattribute{\glscategorylabel}{firstshortaccess}{true}%
6394 {%
6395 \eappto\ExtraCustomAbbreviationFields{firstaccess={\@gls@shortaccess},}%
6396 }%
6397 {}%
6398 }%
6399 {}%
6400 \ifdefempty\@gls@firstpluralaccess
6401 {%
6402 \glsifcategoryattribute{\glscategorylabel}{firstshortaccess}{true}%
6403 {%
6404 \eappto\ExtraCustomAbbreviationFields{%
6405 firstpluralaccess={\@gls@shortaccesspl},%
6406 }%
6407 }%
6408 {}%
6409 }%
6410 {}%
6411 }

```

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 `\glstr<category><field>accsupp` if not already defined.

provideaccsuppcmd

```

6412 \newcommand*\glstrprovideaccsuppcmd[2]{%
6413 \ifcsundef{glstr#1#2accsupp}%
6414 {\csdef{glstr#1#2accsupp}{\glsshortaccsupp}}%
6415 {}%
6416 }

```

SetNoLongAttrs For styles where the name, first and text are just the abbreviation.

```

6417 \newcommand*\glstrAccSuppAbbrSetNoLongAttrs[1]{%
6418 \glssetcategoryattribute{#1}{nameshortaccess}{true}%
6419 \glssetcategoryattribute{#1}{firstshortaccess}{true}%
6420 \glssetcategoryattribute{#1}{textshortaccess}{true}%
6421 \glstrprovideaccsuppcmd{#1}{name}%
6422 \glstrprovideaccsuppcmd{#1}{first}%
6423 \glstrprovideaccsuppcmd{#1}{firstpl}%
6424 \glstrprovideaccsuppcmd{#1}{text}%
6425 \glstrprovideaccsuppcmd{#1}{plural}%
6426 }

```

FirstLongAttrs For styles where the name and text are just the abbreviation. The first form may just be long or may be short and long.

```

6427 \newcommand*{\glxtrAccSuppAbbrSetFirstLongAttrs}[1]{%
6428 \glsetcategoryattribute{#1}{nameshortaccess}{true}%
6429 \glsetcategoryattribute{#1}{textshortaccess}{true}%
6430 \glxtrprovideaccsuppcmd{#1}{name}%
6431 \glxtrprovideaccsuppcmd{#1}{text}%
6432 \glxtrprovideaccsuppcmd{#1}{plural}%
6433 }

```

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.

```

6434 \newcommand*{\glxtrAccSuppAbbrSetTextShortAttrs}[1]{%
6435 \glsetcategoryattribute{#1}{textshortaccess}{true}%
6436 \glxtrprovideaccsuppcmd{#1}{text}%
6437 \glxtrprovideaccsuppcmd{#1}{plural}%
6438 }

```

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.

```

6439 \newcommand*{\glxtrAccSuppAbbrSetNameShortAttrs}[1]{%
6440 \glsetcategoryattribute{#1}{nameshortaccess}{true}%
6441 \glxtrprovideaccsuppcmd{#1}{name}%
6442 }

```

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.

```

6443 \newcommand*{\glxtrAccSuppAbbrSetNameLongAttrs}[1]{%
6444 \glsetcategoryattribute{#1}{firstshortaccess}{true}%
6445 \glsetcategoryattribute{#1}{textshortaccess}{true}%
6446 \glxtrprovideaccsuppcmd{#1}{first}%
6447 \glxtrprovideaccsuppcmd{#1}{firstpl}%
6448 \glxtrprovideaccsuppcmd{#1}{text}%
6449 \glxtrprovideaccsuppcmd{#1}{plural}%
6450 }

```

End of if accsupp part

```

6451 }
6452 {

```

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

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

```

6453 \newcommand*{\glsaccessname}[1]{\glsetentryname{#1}}

```

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

```

6454 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

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

```

6455 \newcommand*{\GLSaccessname}[1]{%
6456 \protect\mfirstucMakeUppercase{\glentryname{#1}}}
```

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

```

6457 \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.

```

6458 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}
```

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

```

6459 \newcommand*{\GLSAccessstext}[1]{%
6460 \protect\mfirstucMakeUppercase{\glentrytext{#1}}}
```

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

```

6461 \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.

```

6462 \newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}
```

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

```

6463 \newcommand*{\GLSAccessplural}[1]{%
6464 \protect\mfirstucMakeUppercase{\glentryplural{#1}}}
```

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

```

6465 \newcommand*{\glsaccessfirst}[1]{\glentryfirst{#1}}
```

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

```

6466 \newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}
```

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

```

6467 \newcommand*{\GLSAccessfirst}[1]{%
6468 \protect\mfirstucMakeUppercase{\glentryfirst{#1}}}
```

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

```

6469 \newcommand*{\glsaccessfirstplural}[1]{\glentryfirstplural{#1}}
```

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

```

6470 \newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}
```

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

```

6471 \newcommand*{\GLScessfirstplural}[1]{%
6472 \protect\mfirstucMakeUppercase{\glentryfirstplural{#1}}}
```


`\glsaccesssymbol` Display the symbol value (no link and no check for existence).
6473 `\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.
6474 `\newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}`

`\GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.
6475 `\newcommand*{\GLSaccesssymbol}[1]{%`
6476 `\protect\mfirstucMakeUppercase{\glsentrysymbol{#1}}}`

`\glsaccesssymbolplural` Display the symbolplural value (no link and no check for existence).
6477 `\newcommand*{\glsaccesssymbolplural}[1]{\glsentrysymbolplural{#1}}`

`\Glsaccesssymbolplural` Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.
6478 `\newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}`

`\GLSaccesssymbolplural` Display the symbolplural value (no link and no check for existence). converted to upper case.
6479 `\newcommand*{\GLSaccesssymbolplural}[1]{%`
6480 `\protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}`

`\glsaccessdesc` Display the desc value (no link and no check for existence).
6481 `\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.
6482 `\newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence). converted to upper case.
6483 `\newcommand*{\GLSaccessdesc}[1]{%`
6484 `\protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}`

`\glsaccessdescplural` Display the descplural value (no link and no check for existence).
6485 `\newcommand*{\glsaccessdescplural}[1]{\glsentrydescplural{#1}}`

`\Glsaccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.
6486 `\newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}`

`\GLSaccessdescplural` Display the descplural value (no link and no check for existence). converted to upper case.
6487 `\newcommand*{\GLSaccessdescplural}[1]{%`
6488 `\protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}`

`\glsaccessshort` Display the short form (no link and no check for existence).
6489 `\newcommand*{\glsaccessshort}[1]{\glsentryshort{#1}}`

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

6490 `\newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}`

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

6491 `\newcommand*{\GLSaccessshort}[1]{%`
6492 `\protect\mfirstucMakeUppercase{\glentryshort{#1}}}`

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

6493 `\newcommand*{\laccessshortpl}[1]{\glentryshortpl{#1}}`

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

6494 `\newcommand*{\Laccessshortpl}[1]{\Glsentryshortpl{#1}}`

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

6495 `\newcommand*{\GLSaccessshortpl}[1]{%`
6496 `\protect\mfirstucMakeUppercase{\glentryshortpl{#1}}}`

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

6497 `\newcommand*{\glsaccesslong}[1]{\glentrylong{#1}}`

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

6498 `\newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}`

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

6499 `\newcommand*{\GLSaccesslong}[1]{%`
6500 `\protect\mfirstucMakeUppercase{\glentrylong{#1}}}`

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

6501 `\newcommand*{\glsaccesslongpl}[1]{\glentrylongpl{#1}}`

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

6502 `\newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}`

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

6503 `\newcommand*{\GLSaccesslongpl}[1]{%`
6504 `\protect\mfirstucMakeUppercase{\glentrylongpl{#1}}}`

`@initaccesskeys` This does nothing if there's no accessibility support.

6505 `\newcommand*{\@gls@initaccesskeys}{}`

`@default@access` This does nothing if there's no accessibility support.

6506 `\newcommand*{\@gls@setup@default@access}{}`

`rSetNoLongAttrs` This does nothing if there's no accessibility support.

6507 `\newcommand*{\glsxtrAccSuppAbbrSetNoLongAttrs}[1]{}`

tFirstLongAttrs This does nothing if there's no accessibility support.
6508 \newcommand*{\glxtrAccSuppAbbrSetFirstLongAttrs}[1]{}
tTextShortAttrs This does nothing if there's no accessibility support.
6509 \newcommand*{\glxtrAccSuppAbbrSetTextShortAttrs}[1]{}
tNameShortAttrs This does nothing if there's no accessibility support.
6510 \newcommand*{\glxtrAccSuppAbbrSetNameShortAttrs}[1]{}
etNameLongAttrs This does nothing if there's no accessibility support.
6511 \newcommand*{\glxtrAccSuppAbbrSetNameLongAttrs}[1]{}
End of else part
6512 }

1.6 Categories

\glscategory Add a new storage key that can be used to indicate a category. The default category is general.
6513 \glssaddstoragekey{category}{general}{\glscategory}
\glisifcategory Convenient shortcut to determine if an entry has the given category.
6514 \newcommand{\glisifcategory}[4]{%
6515 \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
6516 }

Categories can have attributes.

categoryattribute

```
\glissetcategoryattribute{<category>}{<attribute-label>}{<value>}
```

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

```
6517 \newcommand*{\glissetcategoryattribute}[3]{%  

6518 \csdef{@glxtr@categoryattr@@#1@#2}{#3}%  

6519 }
```

categoryattribute

```
\glsggetcategoryattribute{<category>}{<attribute-label>}
```

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

```
6520 \newcommand*{\glsggetcategoryattribute}[2]{%  

6521 \csuse{@glxtr@categoryattr@@#1@#2}%  

6522 }
```

categoryattribute

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

Tests if the category has the given attribute set.

```
6523 \newcommand*{\glshascategoryattribute}[4]{%
6524   \ifcvoid{@glstr@categoryattr@#1@#2}{#4}{#3}%
6525 }
```

glissetattribute

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

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

```
6526 \newcommand*{\glissetattribute}[3]{%
6527   \glsetcategoryattribute{\glscategory{#1}}{#2}{#3}%
6528 }
```

glsggetattribute

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

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

```
6529 \newcommand*{\glsggetattribute}[2]{%
6530   \glsetcategoryattribute{\glscategory{#1}}{#2}%
6531 }
```

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.

```
6532 \newcommand*{\glshasattribute}[4]{%
6533   \ifglstryexists{#1}%
6534   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
6535   {#4}%
6536 }
```

categoryattribute

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

True if category has the attribute with the given value.

```
6537 \newcommand{\glsifcategoryattribute}[5]{%
6538   \ifcsundef{@glxtr@categoryattr@#1@#2}%
6539   {#5}%
6540   {\ifcsstring{@glxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
6541 }
```

\glsifattribute

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

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

```
6542 \newcommand{\glsifattribute}[5]{%
6543   \ifglentryexists{#1}%
6544   {\glsifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
6545   {#5}%
6546 }
```

Set attributes for the default general category:

```
6547 \glsssetcategoryattribute{general}{regular}{true}
```

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

```
6548 \glsssetcategoryattribute{acronym}{regular}{true}
```

regularcategory Convenient shortcut to add the regular attribute.

```
6549 \newcommand*{\glsssetregularcategory}[1]{%
6550   \glsssetcategoryattribute{#1}{regular}{true}%
6551 }
```

regularcategory

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

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

```
6552 \newcommand{\glsifregularcategory}[3]{%
6553   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
6554 }
```

regularcategory

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

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

```
6555 \newcommand{\glsifnotregularcategory}[3]{%
6556   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
6557 }
```

\glsifregular

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

Short cut to determine if an entry has a regular attribute set to true.

```
6558 \newcommand{\glsifregular}[3]{%
6559   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
6560 }
```

\glsifnotregular

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

Short cut to determine if an entry has a regular attribute set to false.

```
6561 \newcommand{\glsifnotregular}[3]{%
6562   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
6563 }
```

\glsforeachincategory

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

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

```
6564 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
6565   \foralllglossaries[#1]{#3}%
6566   {%
6567     \forglsentries[#3]{#4}%
6568     {%
6569       \glsifcategory{#4}{#2}{#5}{}%
6570     }%
6571   }%
6572 }
```

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.

```
6573 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%  
6574   \forallglossaries[#1]{#4}%  
6575   {%  
6576     \forallsentries[#4]{#5}%  
6577     {%  
6578       \glsifattribute{#5}{#2}{#3}{#6}{}%  
6579     }%  
6580   }%  
6581 }
```

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

```
6582 \ifdef\newterm  
6583 {%
```

`\newterm`

```
6584   \renewcommand*{\newterm}[2][ ]{%  
6585     \newglossaryentry{#2}%  
6586     {type={index},category=index,name={#2},%  
6587     description={\glstrpostdescription\nopostdesc},#1}%  
6588   }
```

Indexed terms are regular by default.

```
6589   \glsssetcategoryattribute{index}{regular}{true}
```

trpostdescindex

```
6590   \newcommand*{\glstrpostdescindex}{}  
  
6591 }  
6592 {}
```

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`.

```
6593 \ifdef\printsymbols  
6594 {%
```

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

```
6595 \newcommand*{\glsxtrnewsymbol}[3][]{%
6596   \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
6597 }
```

Symbols are regular by default.

```
6598 \glssetcategoryattribute{symbol}{regular}{true}
```

`rpostdescsymbol`

```
6599 \newcommand*{\glsxtrpostdescsymbol}{}
6600 }
6601 {}
```

Similar for the numbers option.

```
6602 \ifdef\printnumbers
6603 {%
```

`glsxtrnewnumber`

```
6604 \ifdef\printnumbers
6605 \newcommand*{\glsxtrnewnumber}[3][]{%
6606   \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
6607 }
```

Numbers are regular by default.

```
6608 \glssetcategoryattribute{number}{regular}{true}
```

`rpostdescnumber`

```
6609 \newcommand*{\glsxtrpostdescnumber}{}
6610 }
6611 {}
```

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

```
6612 \newcommand*{\glsxtrsetcategory}[2]{%
6613   \@for\@glsxtr@label:=#1\do
6614   {%
6615     \glsfieldxddef{\@glsxtr@label}{category}{#2}%
6616   }%
6617 }
```

`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.

```
6618 \newcommand*{\glsxtrsetcategoryforall}[2]{%
6619   \forallglossaries[#1]{\@glsxtr@type}{%
6620     \forglentries[\@glsxtr@type]{\@glsxtr@label}%
```



```

6621   {%
6622     \glsfieldxddef{\@glsxtr@label}{category}{#2}%
6623   }%
6624 }%
6625 }

```

rfieldtitlecase

`\glsxtrfieldtitlecase{<label>}{<field>}`

Apply title casing to the contents of the given field.

```

6626 \newcommand*{\glsxtrfieldtitlecase}[2]{%
6627   \expandafter\glsxtrfieldtitlecasecs\expandafter
6628   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
6629 }

```

fieldtitlecasecs The command used by `\glsxtrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```

6630 \newcommand*{\glsxtrfieldtitlecasecs}[1]{\xcapitalisewords{#1}}

```

Provide a convenient way to modify glossary styles without having to define a new style just to convert the first letter of fields to upper case.

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

```

6631 \@ifpackageloaded{glossaries-accsupp}
6632 {
6633   \renewcommand*{\glossentrydesc}[1]{%
6634     \glsdoifexistsorwarn{#1}%
6635     {%
6636       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossdescfont` attribute to determine the font applied.

```

6637   \glshasattribute{#1}{glossdescfont}%
6638   {%
6639     \protected@edef\@glsxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
6640     \ifcsdef{\@glsxtr@attrval}%
6641     {%
6642       \letcs{\@glsxtr@glossdescfont}{\@glsxtr@attrval}%
6643     }%
6644     {%
6645       \GlossariesExtraWarning{Unknown control sequence name
6646         ‘\@glsxtr@attrval’ supplied in glossdescfont attribute
6647         for entry ‘#1’. Ignoring}%
6648       \let\@glsxtr@glossdescfont\@firstofone
6649     }%
6650   }%
6651   {\let\@glsxtr@glossdescfont\@firstofone}%

```

```

6652 \glsifattribute{#1}{glossdesc}{firstuc}%
6653 {%
6654 \@glsxtr@glossdescfont{\Glsaccessdesc{#1}}%
6655 }%
6656 {%
6657 \glsifattribute{#1}{glossdesc}{title}%
6658 {%
6659 \@glsxtr@do@titlecaps@warn
6660 \glsdescriptionaccessdisplay
6661 {%
6662 \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
6663 }%
6664 {#1}%
6665 }%
6666 {%
6667 \@glsxtr@glossdescfont{\glsaccessdesc{#1}}%
6668 }%
6669 }%
6670 }%
6671 }
6672 }
6673 {
6674 \renewcommand*{\glossentrydesc}[1]{%
6675 \glsdoifexistsorwarn{#1}%
6676 {%
6677 \glssetabbrvfmt{\glscategory{#1}}%
6678 \glsattribute{#1}{glossdescfont}%
6679 }%

6680 \protected@edef\@glsxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
6681 \ifcsdef{\@glsxtr@attrval}%
6682 {%
6683 \letcs{\@glsxtr@glossdescfont}{\@glsxtr@attrval}%
6684 }%
6685 {%
6686 \GlossariesExtraWarning{Unknown control sequence name
6687 '\@glsxtr@attrval' supplied in glossdescfont attribute
6688 for entry '#1'. Ignoring}%
6689 \let\@glsxtr@glossdescfont\@firstofone
6690 }%
6691 }%
6692 {\let\@glsxtr@glossdescfont\@firstofone}%
6693 \glsifattribute{#1}{glossdesc}{firstuc}%
6694 {%
6695 \@glsxtr@glossdescfont{\Glsentrydesc{#1}}%
6696 }%
6697 {%
6698 \glsifattribute{#1}{glossdesc}{title}%
6699 {%
6700 \@glsxtr@do@titlecaps@warn

```

```

6701      \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}}%
6702      }%
6703      {%
6704      \@glsxtr@glossdescfont{\glsentrydesc{#1}}}%
6705      }%
6706      }%
6707      }%
6708  }
6709 }

```

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

```

6710 \@ifpackageloaded{glossaries-accsupp}
6711 {
6712   \renewcommand*{\glossentryname}[1]{%
6713     \@glsdoifexistsorwarn{#1}%
6714     {%
6715       \glssetabbrvfmt{\glscategory{#1}}}%

```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```

6716   \glsattribute{#1}{glossnamefont}%
6717   {%
6718     \protected@edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
6719     \ifcsdef{\@glsxtr@attrval}%
6720     {%
6721       \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
6722       }%
6723       {%
6724         \GlossariesExtraWarning{Unknown control sequence name
6725           ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
6726           for entry ‘#1’. Reverting to default \string\glsnamefont}%
6727         \let\@glsxtr@glossnamefont\glsnamefont
6728         }%
6729       }%
6730     {\let\@glsxtr@glossnamefont\glsnamefont}%
6731     \glsifattribute{#1}{glossname}{firstuc}%
6732     {%
6733       \glsnameaccessdisplay
6734       {%
6735         \@glsxtr@glossnamefont{\Glsentryname{#1}}}%
6736       }%
6737       {#1}%
6738     }%
6739     {%
6740       \glsifattribute{#1}{glossname}{title}%
6741       {%
6742         \@glsxtr@do@titlecaps@warn
6743         \glsnameaccessdisplay
6744         {%
6745           \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}}%

```

```

6746     }%
6747     {#1}%
6748 }%
6749 {%
6750     \glsifattribute{#1}{glossname}{uc}%
6751     {%
6752         \glsnameaccessdisplay
6753     }%

```

Hide the label from the upper-casing command.

```

6754         \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
6755         \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
6756     }%
6757     {#1}%
6758 }%
6759 {%
6760     \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
6761     \glsnameaccessdisplay
6762     {%
6763         \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
6764     }%
6765     {#1}%
6766 }%
6767 }%
6768 }%

```

Do post-name hook:

```

6769     \glsxtrpostnamehook{#1}%
6770 }%
6771 }
6772 }
6773 {
6774     \renewcommand*{\glossentryname}[1]{%
6775         \@glsdoifexistsorwarn{#1}%
6776     }%
6777     \glssetabbrvfmt{\glscategory{#1}}%
6778     \glshasattribute{#1}{glossnamefont}%
6779     {%
6780         \protected@edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
6781         \ifcsdef{\@glsxtr@attrval}%
6782         {%
6783             \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
6784         }%
6785         {%
6786             \GlossariesExtraWarning{Unknown control sequence name
6787             '\@glsxtr@attrval' supplied in glossnamefont attribute
6788             for entry '#1'. Reverting to default \string\glsnamefont}%
6789             \let\@glsxtr@glossnamefont\glsnamefont
6790         }%
6791     }%

```

```

6792      {\let\@glxtr@glossnamefont\glsnamefont}%
6793      \glsifattribute{#1}{glossname}{firstuc}%
6794      {%
6795        \@glxtr@glossnamefont{\Glsentryname{#1}}%
6796      }%
6797      {%
6798        \glsifattribute{#1}{glossname}{title}%
6799        {%
6800          \@glxtr@do@titlecaps@warn
6801          \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
6802        }%
6803        {%
6804          \glsifattribute{#1}{glossname}{uc}%
6805          {%

```

Hide the label from the upper-casing command.

```

6806      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
6807      \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
6808      }%
6809      {%

```

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.

```

6810      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
6811      \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
6812      }%
6813      }%
6814      }%

```

Do post-name hook.

```

6815      \glxtrpostnamehook{#1}%
6816      }%
6817    }
6818 }

```

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

```

6819 \@ifpackageloaded{glossaries-accsupp}
6820 {
6821   \renewcommand*{\Glossentryname}[1]{%
6822     \@glsdoifexistsorwarn{#1}%
6823     {%
6824       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```

6825     \glshasattribute{#1}{glossnamefont}%
6826     {%
6827       \protected@edef\@glxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
6828       \ifcsdef{\@glxtr@attrval}%
6829       {%
6830         \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%

```

```

6831     }%
6832     {%
6833         \GlossariesExtraWarning{Unknown control sequence name
6834         ‘\@glxtr@attrval’ supplied in glossnamefont attribute
6835         for entry ‘#1’. Reverting to default \string\glsnamefont}%
6836         \let\@glxtr@glossnamefont\glsnamefont
6837     }%
6838 }%
6839 {\let\@glxtr@glossnamefont\glsnamefont}%
6840 \glsnameaccessdisplay
6841 {%
6842     \@glxtr@glossnamefont{\Glsentryname{#1}}%
6843 }%
6844 {#1}%

```

Do post-name hook:

```

6845     \glxtrpostnamehook{#1}%
6846 }%
6847 }
6848 }
6849 {
6850     \renewcommand*{\Glossentryname}[1]{%
6851         \@glsdoifexistsorwarn{#1}%
6852         {%
6853             \glssetabbrvfmt{\glscategory{#1}}%
6854             \glsattribute{#1}{glossnamefont}%
6855         }%
6856         \protected@edef\@glxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
6857         \ifcsdef\@glxtr@attrval{%
6858             {%
6859                 \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
6860             }%
6861             {%
6862                 \GlossariesExtraWarning{Unknown control sequence name
6863                 ‘\@glxtr@attrval’ supplied in glossnamefont attribute
6864                 for entry ‘#1’. Reverting to default \string\glsnamefont}%
6865                 \let\@glxtr@glossnamefont\glsnamefont
6866             }%
6867         }%
6868         {\let\@glxtr@glossnamefont\glsnamefont}%
6869         \@glxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

6870     \glxtrpostnamehook{#1}%
6871 }%
6872 }
6873 }

```

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.

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

```
6874 \newcommand*{\glxtrpostnamehook}[1]{%
6875   \let\@glsnumberformat\@glxtr@defaultnumberformat
6876   \glxtrdoautoindexname{#1}{indexname}%
```

Allow additional code regardless of category:

```
6877   \glsextrapostnamehook{#1}%
```

Allow categories to hook in here.

```
6878   \csuse{glxtrpostname\glscategory{#1}}%
6879 }
```

trapostnamehook

```
6880 \newcommand*{\glsextrapostnamehook}[1]{}%
```

\glsdefpostname Provide a convenient command for defining the post-name hook for the given category.

```
6881 \newcommand*{\glsdefpostname}[2]{%
6882   \csdef{glxtrpostname#1}{#2}%
6883 }
```

etaccessdisplay

```
6884 \ifpackageloaded{glossaries-accsupp}
6885 {
6886   \newcommand*{\glxtr@setaccessdisplay}[1]{%
6887     \ifcsdef{gls#1accessdisplay}%
6888       {\letcs\@glxtr@accessdisplay{gls#1accessdisplay}}%
6889       {%
```

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.

```
6890     \protected@edef\@gls@thisval{#1}%
6891     \@for\@gls@map:=\@gls@keymap\do{%
6892       \protected@edef\@this@key{\expandafter\@secondoftwo\@gls@map}%
6893       \ifdefequal{\@this@key}{\@gls@thisval}%
6894       {%
6895         \protected@edef\@gls@thisval{\expandafter\@firstoftwo\@gls@map}%
6896         \@endfortrue
6897       }%
6898     }%
6899   }%
6900   \ifcsdef{gls\@gls@thisval accessdisplay}%
6901     {\letcs\@glxtr@accessdisplay{gls\@gls@thisval accessdisplay}}%
6902     {\let\@glxtr@accessdisplay\@firstoftwo}%
6903   }%
6904 }
6905 }
6906 {%
```

```

6907 \newcommand*{\glxtr@setaccessdisplay}[1]{%
6908 \let\@glxtr@accessdisplay\@firstoftwo}
6909 }

```

sentrynameother Provide a command that works like \glossentryname but accesses a different field (which must be supplied using its internal field label).

```

6910 \newrobustcmd*{\glossentrynameother}[2]{%
6911 \@glxtr@doifexistsorwarn{#1}%
6912 {%

```

Accessibility support:

```

6913 \glxtr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

6914 \glxtr@setabbrfmt{\glscategory{#1}}%
6915 \glxtr@hasattribute{#1}{glossnamefont}%
6916 {%
6917 \protected@edef\@glxtr@attrval{\glxtr@getattribute{#1}{glossnamefont}}%
6918 \ifcsdef{\@glxtr@attrval}%
6919 {%
6920 \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
6921 }%
6922 {%
6923 \GlossariesExtraWarning{Unknown control sequence name
6924 '\@glxtr@attrval' supplied in glossnamefont attribute
6925 for entry '#1'. Reverting to default \string\glxtr@glossnamefont}%
6926 \let\@glxtr@glossnamefont\glxtr@glossnamefont
6927 }%
6928 }%
6929 {\let\@glxtr@glossnamefont\glxtr@glossnamefont}%
6930 \glxtr@ifattribute{#1}{glossname}{firstuc}%
6931 {%
6932 \glxtr@accessdisplay
6933 {\@glxtr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
6934 {#1}%
6935 }%
6936 {%
6937 \glxtr@ifattribute{#1}{glossname}{title}%
6938 {%
6939 \glxtr@do@titlecaps@warn
6940 \glxtr@accessdisplay
6941 {\@glxtr@glossnamefont{\glxtr@field@titlecase{#1}{#2}}}%
6942 {#1}%
6943 }%
6944 {%
6945 \glxtr@ifattribute{#1}{glossname}{uc}%
6946 {%
6947 \letcs{\glo@name}{\glo@\glxtr@detoklabel{#1}{#2}}%
6948 \glxtr@accessdisplay
6949 {\@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%

```



```

6950      {#1}%
6951    }%
6952  {%
6953    \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@#2}%
6954    \@glsxtr@accessdisplay
6955    {\expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}}%
6956    {#1}%
6957  }%
6958 }%
6959 }%

```

Do post-name hook.

```

6960    \glsxtrpostnamehook{#1}%
6961  }%
6962 }

```

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

```

6963 \newif\if@glsxtr@format@override
6964 \@glsxtr@format@overridefalse

```

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

`xFormatOverride`

```

6965 \@ifpackageloaded{hyperref}
6966 {

```

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

```

6967   \ifHy@hyperindex
6968     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
6969       \@glsxtr@format@override true
6970       \appto\theindex{\let\glsnumber\@firstofone}%
6971     }
6972   \else
6973     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
6974       \@glsxtr@format@override true
6975       \appto\theindex{\let\glsnumber\hyperpage}%
6976     }
6977   \fi
6978 }
6979 {
6980   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
6981     \@glsxtr@format@override true
6982   }
6983 }
6984 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

`doautoindexname`

```

6985 \newcommand*{\glsxtrdoautoindexname}[2]{%

```

```

6986 \glshasattribute{#1}{#2}%
6987 {%

```

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.

```

6988 \@glxtr@autoindex@setname{#1}%

```

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

```

6989 \protected@edef\@glxtr@attrval{\glsggetattribute{#1}{#2}}%
6990 \if@glxtr@format@override

6991 \ifx\@glsnumberformat\@glxtr@defaultnumberformat
6992 \else
6993 \let\@glxtr@attrval\@glsnumberformat
6994 \fi
6995 \fi

6996 \ifdefstring{\@glxtr@attrval}{true}%
6997 {}%
6998 {\protected@eappto\@glo@name{\@glxtr@autoindex@encap\@glxtr@attrval}}%
6999 \expandafter\glxtrautoindex\expandafter{\@glo@name}%
7000 }%
7001 {}%
7002 }

```

glxtrautoindex

```

7003 \newcommand*{\glxtrautoindex}{\index}

```

xtrautoindexesc

```

7004 \newcommand{\glxtrautoindexesc}{%
7005 \@gls@checkmkidxchars\@glo@sort
7006 \@glxtr@autoindex@doextra@esc\@glo@sort
7007 }

```

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

```

7008 \newcommand*{\@glxtr@autoindex@setname}[1]{%
7009 \protected@edef\@glo@name{\glxtrautoindexentry{#1}}%
7010 \glxtrautoindexassignsort{\@glo@sort}{#1}%
7011 \glxtrautoindexesc
7012 \epreto\@glo@name{\@glo@sort\@glxtr@autoindex@at}%
7013 }

```

rautoindexentry Command used for the actual part when auto-indexing.

```

7014 \newcommand*{\glxtrautoindexentry}[1]{\string\glstryname{#1}}

```

indexassignsort Used to assign the sort value when auto-indexing.

```

7015 \newcommand*{\glxtrautoindexassignsort}[2]{%
7016 \glsletentryfield{#1}{#2}{sort}%
7017 }

```

dex@doextra@esc

```
7018 \newcommand*{\@glsxtr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
7019 \ifx\@glsxtr@autoindex@esc\@gls@quotechar
```

```
7020 \else
```

```
7021 \def\@gls@checkedmkidx{}%
```

```
7022 \edef\@@glsxtr@checkspch{%
```

```
7023 \noexpand\@glsxtr@autoindex@escquote\expandonce{#1}%
```

```
7024 \noexpand\@empty\@glsxtr@autoindex@esc\noexpand\@nnil
```

```
7025 \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%
```

```
7026 \@@glsxtr@checkspch
```

```
7027 \let#1\@gls@checkedmkidx\relax
```

```
7028 \fi
```

Escape actual character unless it has already been escaped.

```
7029 \ifx\@glsxtr@autoindex@at\@gls@actualchar
```

```
7030 \else
```

```
7031 \def\@gls@checkedmkidx{}%
```

```
7032 \edef\@@glsxtr@checkspch{%
```

```
7033 \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
```

```
7034 \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
```

```
7035 \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
```

```
7036 \@@glsxtr@checkspch
```

```
7037 \let#1\@gls@checkedmkidx\relax
```

```
7038 \fi
```

Escape level character unless it has already been escaped.

```
7039 \ifx\@glsxtr@autoindex@level\@gls@levelchar
```

```
7040 \else
```

```
7041 \def\@gls@checkedmkidx{}%
```

```
7042 \edef\@@glsxtr@checkspch{%
```

```
7043 \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
```

```
7044 \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
```

```
7045 \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
```

```
7046 \@@glsxtr@checkspch
```

```
7047 \let#1\@gls@checkedmkidx\relax
```

```
7048 \fi
```

Escape encap character unless it has already been escaped.

```
7049 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
```

```
7050 \else
```

```
7051 \def\@gls@checkedmkidx{}%
```

```
7052 \edef\@@glsxtr@checkspch{%
```

```
7053 \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
```

```
7054 \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
```

```
7055 \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
```

```
7056 \@@glsxtr@checkspch
```

```
7057 \let#1\@gls@checkedmkidx\relax
```

```
7058 \fi
```

```
7059 }
```

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`.

```
7060 \newcommand*{\@glxtr@autoindex@at}{}

```

`\trSetActualChar` Set the actual character.

```
7061 \newcommand*{\GlsXtrSetActualChar}[1]{%
7062   \gdef\@glxtr@autoindex@at{#1}%
7063   \def\@glxtr@autoindex@escat##1##2##3\@glxtr@endescspch{%
7064     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escat}{##1}{##2}{##3}%
7065   }%
7066 }
7067 \@onlypreamble\GlsXtrSetActualChar
7068 \makeatother
7069 \GlsXtrSetActualChar{@}
7070 \makeatletter

```

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

```
7071 \newcommand*{\@glxtr@autoindex@encap}{}

```

`\XtrSetEncapChar` Set the encap character.

```
7072 \newcommand*{\GlsXtrSetEncapChar}[1]{%
7073   \gdef\@glxtr@autoindex@encap{#1}%
7074   \def\@glxtr@autoindex@escencap##1##2##3\@glxtr@endescspch{%
7075     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escencap}{##1}{##2}{##3}%
7076   }%
7077 }
7078 \GlsXtrSetEncapChar{|}
7079 \@onlypreamble\GlsXtrSetEncapChar

```

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

```
7080 \newcommand*{\@glxtr@autoindex@level}{}

```

`\XtrSetLevelChar` Set the encap character.

```
7081 \newcommand*{\GlsXtrSetLevelChar}[1]{%
7082   \gdef\@glxtr@autoindex@level{#1}%
7083   \def\@glxtr@autoindex@esclevel##1##2##3\@glxtr@endescspch{%
7084     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@esclevel}{##1}{##2}{##3}%
7085   }%
7086 }
7087 \GlsXtrSetLevelChar{!}
7088 \@onlypreamble\GlsXtrSetLevelChar

```

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

```
7089 \newcommand*{\@glxtr@autoindex@esc}{"}

```

lsXtrSetEscChar Set the escape character.

```
7090 \newcommand*{\GlsXtrSetEscChar}[1]{%
7091   \gdef\@glxtr@autoindex@esc{#1}%
7092   \def\@glxtr@autoindex@escquote##1##2##3\@glxtr@endescspch{%
7093     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escquote}{##1}{##2}{##3}%
7094   }%
7095 }
7096 \GlsXtrSetEscChar{"}
7097 \@onlypreamble\GlsXtrSetEscChar
```

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

```
7098 \ifdef\actualchar
7099 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
7100 {}
```

Quote character \quotechar:

```
7101 \ifdef\quotechar
7102 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
7103 {}
```

Level character \levelchar:

```
7104 \ifdef\levelchar
7105 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
7106 {}
```

Encap character \encapchar:

```
7107 \ifdef\encapchar
7108 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
7109 {}
```

leto@endescspch

```
7110 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

oindex@esc@spch

<code>\@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}</code>
--

```
7111 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
7112   \@glx@tmpb=\expandafter{\@glx@checkedmkidx}%
7113   \toks@={#3}%
7114   \ifx\@nnil#3\relax
7115     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
7116   \else
7117     \ifx\@nnil#4\relax
7118       \edef\@glx@checkedmkidx{\the\@glx@tmpb\the\toks@}%
7119       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
7120         #4#5\@glxtr@endescspch}%
7121     \else
```

```

7122      \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
7123      \@glsxtr@autoindex@esc#1}%
7124      \def\@@glsxtr@checkspch{#2#5#1\@nnil#1\@glsxtr@endescspch}%
7125      \fi
7126      \fi
7127      \@glsxtr@checkspch
7128 }

```

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

```

7129 \renewcommand*{\Glossentrydesc}[1]{%
7130   \glsdoifexistsorwarn{#1}%
7131   {%
7132     \glssetabbrvfmt{\glscategory{#1}}%
7133     \Glsaccessdesc{#1}%
7134   }%
7135 }

```

\glossentrysymbol Redefine to set the format and accessibility support. Allow for the possibility of being used in a section heading for standalone entry definitions.

```

7136 \ifdef\teorpdfstring
7137 {
7138   \renewcommand*{\glossentrysymbol}[1]{%
7139     \teorpdfstring{\@glossentrysymbol{#1}}{\glseentrypdfsymbol{#1}}%
7140   }
7141 }
7142 {
7143   \renewcommand*{\glossentrysymbol}[1]{\@glossentrysymbol{#1}}
7144 }

```

\glsentrypdfsymbol May be redefined to a field that expands to a value that's more suitable for PDF bookmarks.

```

7145 \newcommand{\glseentrypdfsymbol}[1]{\glsentrysymbol{#1}}

```

\glossentrysymbol There are no case-changing attributes as it's less usual for symbols.

```

7146 \newrobustcmd*{\@glossentrysymbol}[1]{%
7147   \glsdoifexistsorwarn{#1}%
7148   {%
7149     \begingroup
7150     \glssetabbrvfmt{\glscategory{#1}}%
7151     \glshasattribute{#1}{glosssymbolfont}%
7152     {%
7153       \protected@edef\@glsxtr@attrval{\glsggetattribute{#1}{glosssymbolfont}}%
7154       \ifcsdef{\@glsxtr@attrval}%
7155       {%
7156         \letcs{\@glsxtr@glosssymbolfont}{\@glsxtr@attrval}%
7157       }%
7158     }%
7159     \GlossariesExtraWarning{Unknown control sequence name
7160     '\@glsxtr@attrval' supplied in glosssymbolfont attribute
7161     for entry '#1'. Ignoring}%

```

```

7162         \let\@glxtr@glosssymbolfont\@firstofone
7163     }%
7164 }%
7165 {\let\@glxtr@glosssymbolfont\@firstofone}%
7166 \@glxtr@glosssymbolfont{\glsaccesssymbol{#1}}%
7167 \endgroup
7168 }%
7169 }

```

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

```

7170 \renewcommand*{\Glossentrysymbol}[1]{%
7171     \glsdoifexistsorwarn{#1}%
7172     {%
7173         \glssetabbrfmt{\glscategory{#1}}%
7174         \Glsaccesssymbol{#1}%
7175     }%
7176 }

```

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

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

```

7177 \newcommand*{\GlsXtrEnableInitialTagging}{%
7178     \@ifstar\s@glxtr@enabletagging\@glxtr@enabletagging
7179 }
7180 \@onlypreamble\GlsXtrEnableInitialTagging

```

`\r@enabletagging` Starred version undefines command.

```

7181 \newcommand*{\s@glxtr@enabletagging}[2]{%
7182     \undef#2%
7183     \@glxtr@enabletagging{#1}{#2}%
7184 }

```

`\r@enabletagging` Internal command.

```

7185 \newcommand*{\@glxtr@enabletagging}[2]{%
    Set attributes for categories given in the first argument.
7186     \@for\@glxtr@cat:=#1\do
7187     {%
7188         \ifdefempty\@glxtr@cat
7189         {%
7190             {\glssetcategoryattribute{\@glxtr@cat}{tagging}{true}}%
7191         }%
7192         \newrobustcmd*#2[1]{##1}%
7193         \def\@glxtr@taggingcs{#2}%
7194         \renewcommand*\@glxtr@activate@initialtagging{%
7195             \let#2\@glxtr@tag
7196         }%

```

```

7197 \ifundef\@gls@preglossaryhook
7198 {\GlossariesExtraWarning{Initial tagging requires at least
7199   glossaries.sty v4.19 to work correctly}}}%
7200 {}%
7201 }

```

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

```

7202 \ifundef\mfu@checkword@do
7203 {
7204   \newcommand*\mfu@checkword@do[1]{%
7205     \ifdefstring{\mfu@checkword@arg}{#1}%
7206     {%
7207       \let\@mfu@domakefirstuc\@firstofone
7208       \listbreak
7209     }%
7210   }%
7211 }

```

\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.

```

7212 \ifundef\mfu@checkword
7213 {
7214   \newcommand{\@glsxtr@do@titlecaps@warn}{%
7215     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
7216       support not available}%

```

One warning should suffice.

```

7217     \let\@glsxtr@do@titlecaps@warn\relax
7218   }
7219 }
7220 {
7221   \renewcommand*\mfu@checkword[1]{%
7222     \def\mfu@checkword@arg{#1}%
7223     \let\@mfu@domakefirstuc\makefirstuc
7224     \forlistloop\mfu@checkword@do\@mfu@nocaplist
7225   }
7226 }
7227 }
7228 {}% no patch required

```

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

```

7229 \newcommand*\@glsxtr@do@titlecaps@warn{}

```

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

```

7230 \newcommand*\@glsxtr@activate@initialtagging{}

```


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

```
7231 \newrobustcmd*{\@glxtr@tag}[1]{%
7232   \gl@ifattribute{\glscurrententrylabel}{tagging}{true}%
7233   {\@glxtrtagfont{#1}}{#1}%
7234 }
```

`\glxtrtagfont` Used in the glossary.

```
7235 \newcommand*{\glxtrtagfont}[1]{\underline{#1}}
```

`preglossaryhook` This macro was introduced in glossaries version 4.19, so it may not be defined. If it hasn't been defined this feature is unavailable. A check is added for the entry's existence to prevent errors from occurring if the user removes an entry or changes the label, which can interrupt the build process.

```
7236 \ifdef\@gls@preglossaryhook
7237 {
7238   \renewcommand*{\@gls@preglossaryhook}{%
7239     \@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.

```
7240   \ifundef\@glxtr@org@postdescription
7241   {%
7242     \let\@glxtr@org@postdescription\glspostdescription
7243     \renewcommand*{\glspostdescription}{%
7244       \ifglstryexists{\glscurrententrylabel}%
7245       {%
7246         \glxtrpostdescription
7247         \@glxtr@org@postdescription
7248       }%
7249     }%
7250   }%
7251 }%
7252 {}%
```

Enable the options used by `\@glxtrp`:

```
7253   \glossxtrsetpopts
7254 }%
7255 }
7256 {}
```

`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.

```
7257 \newcommand*{\glxtrpostdescription}{%
7258   \csuse{glxtrpostdesc\glscategory{\glscurrententrylabel}}%
7259 }
```

postdescgeneral

```
7260 \newcommand*{\glxtrpostdescgeneral}{}%
```

xtrpostdescterm

```
7261 \newcommand*{\glxtrpostdescterm}{}%
```

postdescacronym

```
7262 \newcommand*{\glxtrpostdescacronym}{}%
```

escabbreviation

```
7263 \newcommand*{\glxtrpostdescabbreviation}{}%
```

\glsdefpostdesc Provide a convenient command for defining the post-description hook for the given category.

```
7264 \newcommand*{\glsdefpostdesc}[2]{%
```

```
7265   \csdef{glxtrpostdesc#1}{#2}%
```

```
7266 }
```

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.

```
7267 \renewcommand*{\glspostlinkhook}{}%
```

```
7268 \ifglssentryexists{glslabel}{\glxtrpostlinkhook}{}%
```

```
7269 }
```

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

```
7270 \newcommand*{\glxtrpostlinkhook}{}%
```

```
7271 \glxtrdiscardperiod{glslabel}%
```

```
7272 {\glxtrpostlinkendsentence}%
```

```
7273 {\glxtrifcustomdiscardperiod
```

```
7274   {\glxtrifperiod{\glxtrpostlinkendsentence}{\glxtrpostlink}}%
```

```
7275   {\glxtrpostlink}%
```

```
7276 }%
```

```
7277 }
```

omdiscardperiod Allow user to provide a custom check. Should expand to #2 if no check is required otherwise expand to #1.

```
7278 \newcommand*{\glxtrifcustomdiscardperiod}[2]{#2}
```

\glxtrpostlink

```
7279 \newcommand*{\glxtrpostlink}{}%
```

```
7280 \csuse{glxtrpostlink\glscategory{glslabel}}%
```

```
7281 }
```

\glsdefpostlink Provide a convenient command for defining the post-link hook for the given category. Doesn't allow an empty argument (which) would overwrite `\glxtrpostlink`.

```
7282 \newcommand*{\glsdefpostlink}[2]{%
```

`\ifthenelse` is used to ensure that the expanded value is tested. (The category label must be fully expandable.)

```
7283 \ifthenelse{\equal{#1}{}}%
7284 {\PackageError{glossaries-extra}
7285   {Invalid empty category label in \string\glsdefpostlink}{}}%
7286 {\csdef{glsxtrpostlink#1}{#2}}%
7287 }
```

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

```
7288 \newcommand*{\glsxtrpostlinkendsentence}{%
7289 \ifcsdef{glsxtrpostlink\glscategory{\glslabel}}
7290 {%
7291 \csuse{glsxtrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
7292 .\spacefactor\sfcode'\. \relax
7293 }%
7294 {%
```

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

```
7295 \spacefactor\sfcode'\. \relax
7296 }%
7297 }
```

`DescOnFirstUse` Provide a command for appending the description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
7298 \newcommand*{\glsxtrpostlinkAddDescOnFirstUse}{%
7299 \glsxtrifwasfirstuse{\space\glsxtrparen{\glsaccessdesc{\glslabel}}}{}%
7300 }
```

`symbolOnFirstUse` Provide a command for appending the symbol (if defined) in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
7301 \newcommand*{\glsxtrpostlinkAddSymbolOnFirstUse}{%
7302 \glsxtrifwasfirstuse
7303 {%
7304 \ifglshassymbol{\glslabel}%
7305 {\space\glsxtrparen{\glsaccesssymbol{\glslabel}}}%
7306 {}}%
7307 }%
7308 {}%
7309 }
```

`SymbolDescOnFirstUse` 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.

```
7310 \newcommand*{\glsxtrpostlinkAddSymbolDescOnFirstUse}{%
7311 \glsxtrifwasfirstuse
7312 {%
7313 \space\glsxtrparen
```

```

7314    {%
7315        \ifglshassymbol{\glslabel}%
7316        {\glsaccesssymbol{\glslabel}, }%
7317        }%
7318        \glsaccessdesc{\glslabel}%
7319    }%
7320 }%
7321 {}%
7322 }

```

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).

```

7323 \newcommand*{\glxtrdiscardperiod}[3]{%
7324   \glxtrifwasfirstuse
7325   {%
7326     \glsifattribute{#1}{retainfirstuseperiod}{true}%
7327     {#3}%
7328     {%
7329       \glsifattribute{#1}{discardperiod}{true}%
7330       {%
7331         \glsifplural
7332         {%
7333           \glsifattribute{#1}{pluraldiscardperiod}{true}%
7334           {\glxtrifperiod{#2}{#3}}%
7335           {#3}%
7336         }%
7337         {%
7338           \glxtrifperiod{#2}{#3}%
7339         }%
7340       }%
7341       {#3}%
7342     }%
7343   }%
7344   {%
7345     \glsifattribute{#1}{discardperiod}{true}%
7346     {%
7347       \glsifplural
7348       {%
7349         \glsifattribute{#1}{pluraldiscardperiod}{true}%
7350         {\glxtrifperiod{#2}{#3}}%
7351         {#3}%
7352       }%
7353       {%
7354         \glxtrifperiod{#2}{#3}%
7355       }%
7356     }%
7357     {#3}%

```

```
7358 }%
7359 }
```

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

```
7360 \newcommand*{\glxtrifperiod}[1]{\new@ifnextchar.\{\@firstoftwo{#1}}}
```

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

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

```
7361 \newcommand*{\glxtr@punclist}{.,:;?!}
```

`\punctuationmark` Add character to punctuation list.

```
7362 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}
```

`\punctuationmarks` Reset the punctuation list.

```
7363 \newcommand*{\glxtrsetpunctuationmarks}[1]{\def\glxtr@punclist{#1}}
```

`\glxtrifpunc`

```
\glxtrifnextpunc{\<true part>}{\<false part>}
```

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

```
7364 \newcommand*{\glxtrifnextpunc}[2]{%
7365   \def\reserved@a{#1}%
7366   \def\reserved@b{#2}%
7367   \futurelet\@glspunc@token\glxtr@ifnextpunc
7368 }
```

`\glxtr@ifnextpunc`

```
7369 \newcommand*{\glxtr@ifnextpunc}{%
7370   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
7371   \reserved@b
7372 }
```

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

```
7373 \newcommand*{\glxtr@ifpunctoken}[1]{%
7374   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
7375 }
```

`\glxtr@ifpunctoken`

```
7376 \def\@glxtr@ifpunctoken#1#2{%
7377   \let\reserved@d=#2%
7378   \ifx\reserved@d\@nnil
7379     \let\glxtr@next\@glxtr@notfoundinlist
7380   \else
```

```

7381 \ifx#1\reserved@d
7382 \let\glstr@next\@glstr@foundinlist
7383 \else
7384 \let\glstr@next\@glstr@ifpunctoken
7385 \fi
7386 \fi
7387 \glstr@next#1%
7388 }

```

tr@foundinlist

```

7389 \def\@glstr@foundinlist#1\@nnil{\@firstoftwo}

```

@notfoundinlist

```

7390 \def\@glstr@notfoundinlist#1{\@secondoftwo}

```

lsxtrdopostpunc

```
\glxtrdopostpunc{<code>}
```

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

```

7391 \newcommand{\glxtrdopostpunc}[1]{%
7392 \glxtrifnextpunc{\@glstr@swaptwo{#1}}{#1}%
7393 }

```

@glstr@swaptwo

```

7394 \newcommand{\@glstr@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`.

```

7395 \define@key{glstrabbrv}{category}{%
7396 \protected@edef\glscategorylabel{#1}%
7397 }

```

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

```

7398 \define@key{glstrabbrv}{shortplural}{%
7399 \def\@glstr@shortpl{#1}%
7400 }

```

Similarly for the long plural form.

```
7401 \define@key{glxstrabbrv}{longplural}{%  
7402   \def\@gls@longpl{#1}%  
7403 }
```

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

\glsshortpltok

```
7404 \newtoks\glsshortpltok
```

\glslongpltok

```
7405 \newtoks\glslongpltok
```

7406 \newcommand*{\@glsxtr@insertdots}[2]{%
7407 \def#1{}%
7408 \@glsxtr@insert@dots#1#2\@nnil
7409 }

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.

\@glsxtr@insert@dots

```
7410 \newcommand*{\@glsxtr@insert@dots}[2]{%  
7411   \ifx\@nnil#2\relax  
7412   \let\@glsxtr@insert@dots@next\@gobble  
7413   \else  
7414     \ifx\relax#2\relax  
7415     \else  
7416       \appto#1{#2.}%  
7417     \fi  
7418     \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots  
7419   \fi  
7420   \@glsxtr@insert@dots@next#1%  
7421 }
```

Similarly provide a way of replacing spaces with \glsxtrwordsep, which first needs to be defined:

\glsxtrwordsep

```
7422 \newcommand*{\glsxtrwordsep}{\space}
```

Each word is marked with

\glsxtrword

```
7423 \newcommand*{\glsxtrword}[1]{#1}
```

tr@markwordseps

```
7424 \newcommand*{\@glxstr@markwordseps}[2]{%
7425   \def#1{%
7426     \@glxstr@mark@wordseps#1#2 \@nnil
7427 }
```

r@mark@wordseps

```
7428 \def\@glxstr@mark@wordseps#1#2 #3{%
7429   \ifdefempty{#1}%
7430     {\def#1{\protect\glxstrword{#2}}}%
7431     {\appto#1{\protect\glxstrwordsep\protect\glxstrword{#2}}}%
7432     \ifx\@nnil#3\relax
7433       \let\@glxstr@mark@wordseps@next\relax
7434     \else
7435       \def\@glxstr@mark@wordseps@next{%
7436         \@glxstr@mark@wordseps#1#3}%
7437     \fi
7438     \@glxstr@mark@wordseps@next
7439 }
```

newabbreviation Define a new generic abbreviation.

```
7440 \newcommand*{\newabbreviation}[4][{}]{%
7441   \glxstr@newabbreviation{#1}{#2}{#3}{#4}%
7442 }
```

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.)

```
7443 \newcommand*{\glxstr@newabbreviation}[4]{%
7444   \glskeylisttok{#1}%
7445   \glslabeltok{#2}%
7446   \glsshorttok{#3}%
7447   \glslongtok{#4}%

```

Save the original short and long values (before attribute settings modify them).

```
7448 \def\glxstrorgshort{#3}%
7449 \def\glxstrorglong{#4}%

```

Provide extra settings for hooks (if modified, this command must end with a comma).

```
7450 \def\ExtraCustomAbbreviationFields{}%

```

Initialise accessibility settings if required.

```
7451 \@gl@initaccesskeys

```

Get the category.

```
7452 \def\glscategorylabel{abbreviation}%

```

Ignore the shortplural and longplural keys.

```
7453 \setkeys*{\glxstrabbrv}[shortplural,longplural]{#1}%

```

Set the abbreviation style.

```
7454 \ifcsdef{\gl@abbrv@current@\glscategorylabel}%
7455   {%

```


Warning should already have been issued.

```
7456 \let\@glxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
7457 \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
7458 \glxtr@applyabrvstyle{\csname @glsabrv@current@glscategorylabel\endcsname}%
7459 \let\GlsXtrWarnDeprecatedAbbrStyle\@glxtr@orgwarndep
7460 }%
7461 {%
```

If no style has been associated with this category, fallback on the style for the abbreviation category.

```
7462 \glxtr@applyabrvstyle{\@glsabrv@current@abbreviation}%
7463 }%
```

Set the default long plural

```
7464 \def\@gl@longpl{#4\glspluralsuffix}%
7465 \let\@gl@default@longpl\@gl@longpl
```

Has the markwords attribute been set?

```
7466 \gl@ifcategoryattribute{\glscategorylabel}{markwords}{true}%
7467 {%
7468 \@glxtr@keywordseps\@gl@long{#4}%
7469 \expandafter\def\expandafter\@gl@longpl\expandafter
7470 {\@gl@long\glspluralsuffix}%
7471 \let\@gl@default@longpl\@gl@longpl
```

Update \glslongtok.

```
7472 \expandafter\glslongtok\expandafter{\@gl@long}%
7473 }%
7474 {}%
```

Has the markshortwords attribute been set? (Not compatible with insertdots.)

```
7475 \gl@ifcategoryattribute{\glscategorylabel}{markshortwords}{true}%
7476 {%
7477 \@glxtr@keywordseps\@gl@short{#3}%
7478 }%
7479 {}%
```

Has the insertdots attribute been set?

```
7480 \gl@ifcategoryattribute{\glscategorylabel}{insertdots}{true}%
7481 {%
7482 \@glxtr@insertdots\@gl@short{#3}%
7483 \appto\@gl@short{\@}%
7484 }%
7485 {\def\@gl@short{#3}}%
7486 }%
```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```
7487 \gl@ifcategoryattribute{\glscategorylabel}{aposplural}{true}%
7488 {%
7489 \expandafter\def\expandafter\@gl@shortpl\expandafter{\@gl@short
7490 'abrvpluralsuffix}%

```

```

7491 }%
7492 {%
    Has the noshortplural attribute been set?
7493     \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
7494     {%
7495         \let\@gls@shortpl\@gls@short
7496     }%
7497     {%
7498         \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
7499             \abbrvpluralsuffix}%
7500     }%
7501 }%

    Update \glsshorttok:
7502     \expandafter\glsshorttok\expandafter{\@gls@short}%

    Hook for further customisation if required:
7503     \glsxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%

    Get the short and long plurals provided by user in optional argument to override defaults, if
    necessary. Ignore the category key (already obtained).
7504     \setkeys*{\glsxtrabbrv}[category]{#1}%

    Save in case required.
7505     \let\@gls@org@longpl\@gls@longpl
7506     \let\@gls@org@shortpl\@gls@shortpl

    Has the plural been explicitly set?
7507     \ifx\@gls@default@longpl\@gls@longpl
7508     \else

        Has the markwords attribute been set?
7509         \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
7510         {%
7511             \expandafter\@glxtr@markwordseps\expandafter\@gls@longpl\expandafter
7512                 {\@gls@longpl}%
7513         }%
7514         {}%
7515     \fi

    Set the plural token registers so the values can be accessed by the abbreviation styles.
7516     \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
7517     \expandafter\glslongpltok\expandafter{\@gls@longpl}%

    Hook for accessibility support (does nothing if glossaries-accsupp hasn't been loaded).
7518     \@gls@setup@default@access

    Do any extra setup provided by hook:
7519     \newabbreviationhook

    Define this entry:
7520     \protected@edef\@do@newglossaryentry{%

```

```

7521 \noexpand\newglossaryentry{\the\glslabeltok}%
7522 {%
7523     type=\glstrabbrvtype,%
7524     category=abbreviation,%
7525     short={\the\glsshorttok},%
7526     shortplural={\the\glsshortpltok},%
7527     long={\the\glslongtok},%
7528     longplural={\the\glslongpltok},%
7529     name={\the\glsshorttok},%
7530     \CustomAbbreviationFields,%

```

Hook may override abbreviation style default settings (this hook must end with a comma if set).

```

7531 \ExtraCustomAbbreviationFields

```

Any explicit fields set in the optional argument override all other settings.

```

7532 \the\glskeylisttok
7533 }%
7534 }%
7535 \@do@newglossaryentry

```

Obtain the type and add it to the list of abbreviations.

```

7536 \@glstr@addabbreviationlist{\glstentrytype{\the\glslabeltok}}%
7537 \GlsXtrPostNewAbbreviation
7538 }

```

evpresetkeyhook Hook for extra stuff in `\newabbreviation`

```

7539 \newcommand*{\glstrnewabbrevpresetkeyhook}[3]{}

```

NewAbbreviation Hook used by abbreviation styles.

```

7540 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

bbreviationhook Hook for use with `\newabbreviation`.

```

7541 \newcommand*{\newabbreviationhook}{}

```

reviationFields

```

7542 \newcommand*{\CustomAbbreviationFields}{}

```

\glstrparen For the parenthetical styles.

```

7543 \newcommand*{\glstrparen}[1]{(#1)}

```

lsxtrfullformat Full format without case change.

```

7544 \newcommand*{\glstrfullformat}[2]{%
7545     \glsfirstlongfont{\glsaccesslong{#1}}#2\glstrfullsep{#1}%
7546     \glstrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
7547 }

```

lsxtrfullformat Full format with case change.

```

7548 \newcommand*{\Glsxtrfullformat}[2]{%

```

```

7549 \glsfirstlongfont{\Glsaccesslong{#1}}#2\glstrfullsep{#1}%
7550 \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
7551 }

```

`\glstrfullplformat` Plural full format without case change.

```

7552 \newcommand*{\glstrfullplformat}[2]{%
7553   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glstrfullsep{#1}%
7554   \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
7555 }

```

`\glstrfullplformat` Plural full format with case change.

```

7556 \newcommand*{\Glsxtrfullplformat}[2]{%
7557   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glstrfullsep{#1}%
7558   \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
7559 }

```

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

```

7560 \newcommand*{\glstrfullsep}[1]{\space}

```

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

`\glstryfullformat` Full format without case change.

```

7561 \newcommand*{\glstryinlinefullformat}{\glstryfullformat}

```

`\glstryfullformat` Full format with case change.

```

7562 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}

```

`\glstryfullplformat` Plural full format without case change.

```

7563 \newcommand*{\glstryinlinefullplformat}{\glstryfullplformat}

```

`\glstryfullplformat` Plural full format with case change.

```

7564 \newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}

```

Redefine `\glstryfull` 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 `\glstrfull` set of commands instead.

`\glstryfull`

```

7565 \renewcommand*{\glstryfull}[1]{\glstryinlinefullformat{#1}{}}

```

`\Glsstryfull`

```

7566 \renewcommand*{\Glsstryfull}[1]{\Glsxtrinlinefullformat{#1}{}}

```

`\glstryfullpl`

```

7567 \renewcommand*{\glstryfullpl}[1]{\glstryinlinefullplformat{#1}{}}

```

`\Glsentryfullpl`

7568 `\renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}`

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

7569 `\newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.

7570 `\newcommand*{\glsabbrvdefaultfont}[1]{\glsabbrvdefaultfont{#1}}`

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use. This is redefined by the abbreviation styles, as appropriate.

7571 `\newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont`

7572 `\newcommand*{\glsabbrvdefaultfont}[1]{#1}`

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.

7573 `\newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}`

`\glslongdefaultfont` Default font changing command used for the long form in commands like `\glsxtrlong`.

7574 `\newcommand*{\glslongdefaultfont}[1]{#1}`

`\glsfirstlongfont` Font changing command used for the long form on first use or in the full format.

7575 `\newcommand*{\glsfirstlongfont}[1]{\glsfirstlongfont{#1}}`

`\glsfirstlongdefaultfont`

7576 `\newcommand*{\glsfirstlongdefaultfont}[1]{\glsfirstlongdefaultfont{#1}}`

`\glsbrvpluralsuffix` Default plural suffix. Allow an alternative default suffix for abbreviations.

7577 `\newcommand*{\glsbrvpluralsuffix}{\glspluralsuffix}`

`\glsbrvpluralsuffix` Default plural suffix.

7578 `\newcommand*{\glsbrvpluralsuffix}{\glsbrvpluralsuffix}`

`\glsxtrfull` Full form (no case-change).

7579 `\newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\@ns@glsxtrfull}`

7580 `\newcommand*{\@ns@glsxtrfull}[2][\@gls@hyp@opt\@ns@glsxtrfull]{\@gls@hyp@opt\@ns@glsxtrfull}`

7581 `\new@ifnextchar[\@gls@hyp@opt\@ns@glsxtrfull]{\@gls@hyp@opt\@ns@glsxtrfull}`

7582 `\@gls@hyp@opt\@ns@glsxtrfull`

7583 }

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

7584 `\def\@glsxtr@full#1#2[#3]{\@gls@hyp@opt\@ns@glsxtrfull{#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).

```

7585 \@glxtr@record{#1}{#2}{glslink}%
7586 \glsdoifexists{#2}%
7587 {%
7588   \glsetabbrvfmt{\glscategory{#2}}%
7589   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7590   \let\gl@ifplural\@secondoftwo
7591   \let\gl@scapscase\@firstofthree
7592   \let\gl@insert\@empty
7593   \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.

```

7594 \glxtrsetupfulldefs
7595 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7596 }%
7597 \glspostlinkhook
7598 }

```

trsetupfulldefs

```

7599 \newcommand*{\glxtrsetupfulldefs}{%
7600   \let\glxtrifwasfirstuse\@firstoftwo
7601 }

```

\Glsxtrfull Full form (first letter uppercase).

```

7602 \newrobustcmd*{\Glsxtrfull}{\@gl@hyp@opt\@ns@Glsxtrfull}
7603 \newcommand*\ns@Glsxtrfull[2][{}]{%
7604   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}%
7605     {\@Glsxtr@full{#1}{#2}[]}%
7606 }

```

\@Glsxtr@full Low-level macro:

```

7607 \def\@Glsxtr@full#1#2[#3]{%
7608   \glsdoifexists{#2}%
7609   {%
7610     \glsetabbrvfmt{\glscategory{#2}}%
7611     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7612     \let\gl@ifplural\@secondoftwo
7613     \let\gl@scapscase\@secondofthree
7614     \let\gl@insert\@empty
7615     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
7616     \glxtrsetupfulldefs
7617     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7618   }%
7619   \glspostlinkhook
7620 }

```

`\GLSxtrfull` Full form (all uppercase).

```
7621 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\@ns@GLSxtrfull}
7622 \newcommand*\ns@GLSxtrfull[2][\%
7623   \new@ifnextchar[\@GLSxtr@full{#1}{#2}]{\%
7624     {\@GLSxtr@full{#1}{#2}[]}\%
7625 }
```

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

```
7626 \def\@GLSxtr@full#1#2[#3]{\%
7627   \glsdoifexists{#2}\%
7628   {\%
7629     \glssetabbrvfmt{\glscategory{#2}}\%
7630     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7631     \let\glsifplural\@secondoftwo
7632     \let\glsupcase\@thirdofthree
7633     \let\glsinsert\@empty
7634     \def\glscustomtext{\mfirstucMakeUppercase{\glsxtrinlinefullformat{#2}{#3}}}\%
7635     \glsxtrsetupfulldefs
7636     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}\%
7637   }\%
7638   \glspostlinkhook
7639 }
```

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

```
7640 \newrobustcmd*{\glsxtrfullpl}{\@gls@hyp@opt\@ns@glsxtrfullpl}
7641 \newcommand*\ns@glsxtrfullpl[2][\%
7642   \new@ifnextchar[\@glsxtr@fullpl{#1}{#2}]{\%
7643     {\@glsxtr@fullpl{#1}{#2}[]}\%
7644 }
```

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

```
7645 \def\@glsxtr@fullpl#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).
7646   \@glsxtr@record{#1}{#2}{\glslink}\%
7647   \glsdoifexists{#2}\%
7648   {\%
7649     \glssetabbrvfmt{\glscategory{#2}}\%
7650     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7651     \let\glsifplural\@firstoftwo
7652     \let\glsupcase\@firstofthree
7653     \let\glsinsert\@empty
7654     \def\glscustomtext{\glsxtrinlinefullplformat{#2}{#3}}\%
7655     \glsxtrsetupfulldefs
7656     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}\%
7657   }\%
7658   \glspostlinkhook
7659 }
```

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

```
7660 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
7661 \newcommand*\ns@Glsxtrfullpl[2][{}]{%
7662   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}]{%
7663     {\@Glsxtr@fullpl{#1}{#2}[]}%
7664 }
```

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

```
7665 \def\@Glsxtr@fullpl#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).
7666   \@glsxtr@record{#1}{#2}{glslink}%
7667   \glsdoifexists{#2}%
7668   {%
7669     \glssetabbrvfmt{\gls@category{#2}}%
7670     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7671     \let\glsifplural\@firstoftwo
7672     \let\gls@scapscase\@secondofthree
7673     \let\glsinsert\@empty
7674     \def\gls@customtext{\Glsxtrinlinefullplformat{#2}{#3}}%
7675     \glsxtrsetupfulldefs
7676     \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}%
7677   }%
7678   \gls@postlinkhook
7679 }
```

`\GLSxtrfullpl` Plural full form (all upper case).

```
7680 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\@ns@GLSxtrfullpl}
7681 \newcommand*\ns@GLSxtrfullpl[2][{}]{%
7682   \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}]{%
7683     {\@GLSxtr@fullpl{#1}{#2}[]}%
7684 }
```

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

```
7685 \def\@GLSxtr@fullpl#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).
7686   \@glsxtr@record{#1}{#2}{glslink}%
7687   \glsdoifexists{#2}%
7688   {%
7689     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7690     \let\glsifplural\@firstoftwo
7691     \let\gls@scapscase\@thirdofthree
7692     \let\glsinsert\@empty
7693     \def\gls@customtext{%
7694       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
7695     \glsxtrsetupfulldefs
```



```

7696 \gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7697 }%
7698 \glspostlinkhook
7699 }

```

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

`\glsxtrshort`

```

7700 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\@ns@glsxtrshort}

```

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

```

7701 \newcommand*{\ns@glsxtrshort}[2] [] {%
7702 \new@ifnextchar[{\@glsxtrshort{#1}{#2}}{\@glsxtrshort{#1}{#2} []}%
7703 }

```

Read in the final optional argument:

```

7704 \def\@glsxtrshort#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).

```

7705 \glsxtr@record{#1}{#2}{glslink}%
7706 \glsdoifexists{#2}%
7707 {%

```

Need to make sure `\glsabbrvfont` is set correctly.

```

7708 \glssetabbrvfmt{\glscategory{#2}}%
7709 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7710 \let\glsxtrifwasfirstuse\@secondoftwo
7711 \let\glsifplural\@secondoftwo
7712 \let\glscapscase\@firstofthree
7713 \let\glsinsert\@empty
7714 \def\glscustomtext{%
7715 \glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
7716 \ifglsxtrininsertinside\else#3\fi
7717 }%
7718 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7719 }%
7720 \glspostlinkhook
7721 }

```

`\Glsxtrshort`

```

7722 \newrobustcmd*{\Glsxtrshort}{\@gls@hyp@opt\@ns@Glsxtrshort}

```

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

```

7723 \newcommand*{\ns@Glsxtrshort}[2] [] {%
7724 \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} []}%
7725 }

```

Read in the final optional argument:

```

7726 \def\@Glsxtrshort#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).

```

7727 \@glsxtr@record{#1}{#2}{glslink}%
7728 \glsdoifexists{#2}%
7729 {%
7730 \glssetabbrvfmt{\glscategory{#2}}%
7731 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7732 \let\glsxtrifwasfirstuse\@secondoftwo
7733 \let\glsifplural\@secondoftwo
7734 \let\glsapscase\@secondofthree
7735 \let\glsinsert\@empty
7736 \def\glscustomtext{%
7737 \glsabbrvfont{\Glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
7738 \ifglsxtrininsertinside\else#3\fi
7739 }%
7740 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
7741 }%
7742 \glspostlinkhook
7743 }

```

\Glsxtrshort

```

7744 \newrobustcmd*{\Glsxtrshort}{\@gls@hyp@opt\ns@Glsxtrshort}

```

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

```

7745 \newcommand*{\ns@Glsxtrshort}[2][{}]{%
7746 \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2}[]}%
7747 }

```

Read in the final optional argument:

```

7748 \def\@Glsxtrshort#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).

```

7749 \@glsxtr@record{#1}{#2}{glslink}%
7750 \glsdoifexists{#2}%
7751 {%
7752 \glssetabbrvfmt{\glscategory{#2}}%
7753 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7754 \let\glsxtrifwasfirstuse\@secondoftwo
7755 \let\glsifplural\@secondoftwo
7756 \let\glsapscase\@thirdofthree
7757 \let\glsinsert\@empty
7758 \def\glscustomtext{%
7759 \mfirstucMakeUppercase
7760 {\glsabbrvfont{\Glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
7761 \ifglsxtrininsertinside\else#3\fi
7762 }%
7763 }%
7764 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
7765 }%

```

```

7766 \glspostlinkhook
7767 }

```

\glsxtrlong

```

7768 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
7769 \newcommand*{\ns@glxtrlong}[2] [] {%
7770   \new@ifnextchar[{\@glxtrlong{#1}{#2}}{\@glxtrlong{#1}{#2} []}%
7771 }

```

Read in the final optional argument:

```

7772 \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).
7773   \@glxtr@record{#1}{#2}{glslink}%
7774   \glsoifexists{#2}%
7775   {%
7776     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7777     \let\glxtrifwasfirstuse\@secondoftwo
7778     \let\glxtrifplural\@secondoftwo
7779     \let\glscapscase\@firstofthree
7780     \let\glxtrinsert\@empty
7781     \def\glscustomtext{%
7782       \glslongfont{\glssaccesslong{#2}\ifglxtrinsertinside#3\fi}%
7783       \ifglxtrinsertinside\else#3\fi
7784     }%
7785     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7786   }%
7787   \glspostlinkhook
7788 }

```

\Glsxtrlong

```

7789 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
7790 \newcommand*{\ns@Glsxtrlong}[2] [] {%
7791   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
7792 }

```

Read in the final optional argument:

```

7793 \def\@Glsxtrlong#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).
7794   \@glxtr@record{#1}{#2}{glslink}%
7795   \glsoifexists{#2}%
7796   {%
7797     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7798     \let\glxtrifwasfirstuse\@secondoftwo

```

```

7799 \let\glsifplural\@secondoftwo
7800 \let\glsifscapscase\@secondofthree
7801 \let\glsinsert\@empty
7802 \def\glscustomtext{%
7803   \glsfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
7804   \ifglstrinsertinside\else#3\fi
7805 }%
7806 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7807 }%
7808 \glspostlinkhook
7809 }

```

\GLSxtrlong

```

7810 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}

```

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

```

7811 \newcommand*{\ns@GLSxtrlong}[2] [] {%
7812   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
7813 }

```

Read in the final optional argument:

```

7814 \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).

```

7815   \@glsxtr@record{#1}{#2}{\glslink}%
7816   \glsdoifexists{#2}%
7817   {%
7818     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7819     \let\glsxtrifwasfirstuse\@secondoftwo
7820     \let\glsifplural\@secondoftwo
7821     \let\glsifscapscase\@thirdofthree
7822     \let\glsinsert\@empty
7823     \def\glscustomtext{%
7824       \mfirstucMakeUppercase
7825       {\glsfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
7826       \ifglstrinsertinside\else#3\fi
7827     }%
7828     }%
7829     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7830   }%
7831   \glspostlinkhook
7832 }

```

Plural short forms:

\glsxtrshortpl

```

7833 \newrobustcmd*{\glsxtrshortpl}{\@gls@hyp@opt\ns@glsxtrshortpl}

```

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

```

7834 \newcommand*{\ns@glsxtrshortpl}[2] [] {%

```

```

7835 \new@ifnextchar[{\@glsxtrshortpl{#1}{#2}}{\@glsxtrshortpl{#1}{#2} []}%
7836 }

```

Read in the final optional argument:

```

7837 \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).

```

7838 \@glsxtr@record{#1}{#2}{glslink}%
7839 \glsdoifexists{#2}%
7840 {%
7841 \glssetabbrvfmt{\glscategory{#2}}%
7842 \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
7843 \let\glsxtrifwasfirstuse\@secondoftwo
7844 \let\glsifplural\@firstoftwo
7845 \let\glscapscase\@firstofthree
7846 \let\glsinsert\@empty
7847 \def\glscustomtext{%
7848 \glsabbrvfont{\glssaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
7849 \ifglsxtrininsertinside\else#3\fi
7850 }%
7851 \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7852 }%
7853 \glspostlinkhook
7854 }

```

\Glsxtrshortpl

```

7855 \newrobustcmd*{\Glsxtrshortpl}{\@glsl@hyp@opt\@ns@Glsxtrshortpl}

```

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

```

7856 \newcommand*{\@ns@Glsxtrshortpl}[2] []{%
7857 \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
7858 }

```

Read in the final optional argument:

```

7859 \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).

```

7860 \@glsxtr@record{#1}{#2}{glslink}%
7861 \glsdoifexists{#2}%
7862 {%
7863 \glssetabbrvfmt{\glscategory{#2}}%
7864 \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
7865 \let\glsxtrifwasfirstuse\@secondoftwo
7866 \let\glsifplural\@firstoftwo
7867 \let\glscapscase\@secondofthree
7868 \let\glsinsert\@empty
7869 \def\glscustomtext{%
7870 \glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
7871 \ifglsxtrininsertinside\else#3\fi

```

```

7872 }%
7873 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7874 }%
7875 \glspostlinkhook
7876 }

```

\GLSxtrshortpl

```

7877 \newrobustcmd*{\GLSxtrshortpl}{\@gls@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
7878 \newcommand*{\ns@GLSxtrshortpl}[2][{}]{%
7879   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2}[]}%
7880 }

```

Read in the final optional argument:

```

7881 \def\@GLSxtrshortpl#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).
7882 \@glsxtr@record{#1}{#2}{glslink}%
7883 \glsdoifexists{#2}%
7884 {%
7885   \glssetabbrvfmt{\glscategory{#2}}%
7886   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7887   \let\glsxtrifwasfirstuse\@secondoftwo
7888   \let\glsifplural\@firstoftwo
7889   \let\glsapscase\@thirdofthree
7890   \let\glsinsert\@empty
7891   \def\glscustomtext{%
7892     \mfirstucMakeUppercase
7893     {\glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
7894     \ifglsxtrininsertinside\else#3\fi
7895   }%
7896   }%
7897   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7898 }%
7899 \glspostlinkhook
7900 }

```

Plural long forms:

\glsxtrlongpl

```

7901 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\ns@glsxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
7902 \newcommand*{\ns@glsxtrlongpl}[2][{}]{%
7903   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2}[]}%
7904 }

```

Read in the final optional argument:

```

7905 \def\@glsxtrlongpl#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).

```

7906 \@glstr@record{#1}{#2}{glslink}%
7907 \glsoifexists{#2}%
7908 {%
7909 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7910 \let\glstrifwasfirstuse\@secondoftwo
7911 \let\gl@ifplural\@firstoftwo
7912 \let\glscapscase\@firstofthree
7913 \let\glinsert\@empty
7914 \def\glscustomtext{%
7915 \glslongfont{\glaccesslongpl{#2}\ifglstrinsertinside#3\fi}%
7916 \ifglstrinsertinside\else#3\fi
7917 }%
7918 \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
7919 }%
7920 \glspostlinkhook
7921 }

```

\Glsxtrlongpl

```

7922 \newrobustcmd*{\Glsxtrlongpl}{\@gl@hyp@opt\@ns@Glsxtrlongpl}

```

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

```

7923 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
7924 \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
7925 }

```

Read in the final optional argument:

```

7926 \def\@Glsxtrlongpl#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).

```

7927 \@glstr@record{#1}{#2}{glslink}%
7928 \glsoifexists{#2}%
7929 {%
7930 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7931 \let\glstrifwasfirstuse\@secondoftwo
7932 \let\gl@ifplural\@firstoftwo
7933 \let\glscapscase\@secondofthree
7934 \let\glinsert\@empty
7935 \def\glscustomtext{%
7936 \glslongfont{\Glsaccesslongpl{#2}\ifglstrinsertinside#3\fi}%
7937 \ifglstrinsertinside\else#3\fi
7938 }%
7939 \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
7940 }%
7941 \glspostlinkhook
7942 }

```

\GLSxtrlongpl

```
7943 \newrobustcmd*{\GLSxtrlongpl}{\@gls@hyp@opt\ns@GLSxtrlongpl}
```

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

```
7944 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
```

```
7945   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
```

```
7946 }
```

Read in the final optional argument:

```
7947 \def\@GLSxtrlongpl#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).

```
7948   \@glsxtr@record{#1}{#2}{glslink}%
```

```
7949   \glsdoifexists{#2}%
```

```
7950   {%
```

```
7951     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
7952     \let\glsxtrifwasfirstuse\@secondoftwo
```

```
7953     \let\glsifplural\@firstoftwo
```

```
7954     \let\glsapscase\@thirdofthree
```

```
7955     \let\glsinsert\@empty
```

```
7956     \def\glscustomtext{%
```

```
7957       \mfirstucMakeUppercase
```

```
7958       {\glsfont{\glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
```

```
7959       \ifglsxtrininsertinside\else#3\fi
```

```
7960     }%
```

```
7961   }%
```

```
7962   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
```

```
7963 }%
```

```
7964 \glspostlinkhook
```

```
7965 }
```

\glssetabbrvfmt Set the current format for the given category (or the abbreviation category if unset).

```
7966 \newcommand*{\glssetabbrvfmt}[1] {%
```

```
7967   \ifcsdef{\glsabbrv@current@#1}%
```

```
7968   {\glsxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
```

```
7969   {\glsxtr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
```

```
7970 }
```

glsuseabbrvfont Provide a way to use the abbreviation font for a given category for arbitrary text.

```
7971 \newrobustcmd*{\glsuseabbrvfont}[2] {{\glssetabbrvfmt{#2}\glsabbrvfont{#1}}}
```

\glsuselongfont Provide a way to use the long font for a given category for arbitrary text.

```
7972 \newrobustcmd*{\glsuselongfont}[2] {{\glssetabbrvfmt{#2}\glsfont{#1}}}
```

sxtrgenabbrvfmt Similar to \glsacfm, but for abbreviations.

```
7973 \newcommand*{\glsxtrgenabbrvfmt} {%
```

```
7974   \ifdefempty\glscustomtext
```

```
7975   {%
```

```
7976     \ifglsused\glslabel
```

```
7977   }%
```


Subsequent use:

7978 \glsifplural
7979 {%

Subsequent plural form:

7980 \glscapscase
7981 {%

Subsequent plural form, don't adjust case:

7982 \glxtrsubsequentplfmt{\glslabel}{\glsinsert}%
7983 }%
7984 {%

Subsequent plural form, make first letter upper case:

7985 \Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%
7986 }%
7987 {%

Subsequent plural form, all caps:

7988 \mfirstucMakeUppercase
7989 {\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%
7990 }%
7991 }%
7992 {%

Subsequent singular form

7993 \glscapscase
7994 {%

Subsequent singular form, don't adjust case:

7995 \glxtrsubsequentfmt{\glslabel}{\glsinsert}%
7996 }%
7997 {%

Subsequent singular form, make first letter upper case:

7998 \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%
7999 }%
8000 {%

Subsequent singular form, all caps:

8001 \mfirstucMakeUppercase
8002 {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%
8003 }%
8004 }%
8005 }%
8006 {%

First use:

8007 \glsifplural
8008 {%

First use plural form:

8009 \glscapscase
8010 {%

First use plural form, don't adjust case:

```
8011      \glxtrfullplformat{\glslabel}{\glsinsert}%
8012      }%
8013      {%
```

First use plural form, make first letter upper case:

```
8014      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
8015      }%
8016      {%
```

First use plural form, all caps:

```
8017      \mfirstucMakeUppercase
8018      {\glxtrfullplformat{\glslabel}{\glsinsert}}%
8019      }%
8020      }%
8021      {%
```

First use singular form

```
8022      \glscapscase
8023      {%
```

First use singular form, don't adjust case:

```
8024      \glxtrfullformat{\glslabel}{\glsinsert}%
8025      }%
8026      {%
```

First use singular form, make first letter upper case:

```
8027      \Glsxtrfullformat{\glslabel}{\glsinsert}%
8028      }%
8029      {%
```

First use singular form, all caps:

```
8030      \mfirstucMakeUppercase
8031      {\glxtrfullformat{\glslabel}{\glsinsert}}%
8032      }%
8033      }%
8034      }%
8035      }%
8036      {%
```

User supplied text.

```
8037      \glscustomtext
8038      }%
8039      }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
8040 \newcommand*{\glxtrsubsequentfmt}[2]{%
8041   \glsabbrvfont{\glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
8042   \ifglxtrininsertinside \else#2\fi
8043 }
8044 \let\glxtrdefaultsubsequentfmt\glxtrsubsequentfmt
```

```

subsequentplfmt Subsequent use format (plural no case change).
8045 \newcommand*{\glxtrsubsequentplfmt}[2]{%
8046   \glsabbrvfont{\glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
8047   \ifglxtrininsertinside \else#2\fi
8048 }
8049 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt

```

```

trsubsequentfmt Subsequent use format (singular, first letter uppercase).
8050 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
8051   \glsabbrvfont{\Glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
8052   \ifglxtrininsertinside \else#2\fi
8053 }
8054 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt

```

```

subsequentplfmt Subsequent use format (plural, first letter uppercase).
8055 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
8056   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
8057   \ifglxtrininsertinside \else#2\fi
8058 }
8059 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt

```

1.7.1 Abbreviation Styles Setup

```

breviationstyle
8060 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
8061   \ifcsundef{@glsabbrv@dispstyle@setup@#2}
8062   {%
8063     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
8064   }%
8065   {%
      Have abbreviations already been defined for this category?
8066     \ifcsstring{@glsabbrv@current@#1}{#2}%
8067     {%
        Style already set.
8068     }%
8069     {%
8070       \def\@glxtr@dostylewarn{%
8071         \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
8072         {%
8073           \def\@glxtr@dostylewarn{\GlossariesWarning{Abbreviation
8074             style has been switched \MessageBreak
8075             for category ‘#1’, \MessageBreak
8076             but there have already been entries \MessageBreak
8077             defined for this category. Unwanted \MessageBreak
8078             side-effects may result}}%
8079           \@endfortrue
8080         }%
8081         \@glxtr@dostylewarn

```

Set up the style for the given category.

```
8082      \csdef{@glsabbrv@current@#1}{#2}%
8083      \protected@edef\glscategorylabel{#1}%
8084      \glstr@applyabbrvstyle{#2}%
8085      }%
8086  }%
8087 }
```

`\applyabbrvstyle` Apply the abbreviation style without existence check.

```
8088 \newcommand*{\glstr@applyabbrvstyle}[1]{%
8089   \csuse{@glsabbrv@dispstyle@setup@#1}%
8090   \csuse{@glsabbrv@dispstyle@fmts@#1}%
8091 }
```

`\r@applyabbrvfmt` Only apply the style formats.

```
8092 \newcommand*{\glstr@applyabbrvfmt}[1]{%
8093   \csuse{@glsabbrv@dispstyle@fmts@#1}%
8094 }
```

`\newabbreviationstyle` 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.

```
8095 \newcommand*{\newabbreviationstyle}[3]{%
8096   \ifcsdef{@glsabbrv@dispstyle@setup@#1}%
8097   {%
8098     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
8099       defined}{}%
8100   }%
8101   {%
8102     \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
8103     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
8104     #2}%
8105     \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
8106     \renewcommand*{\glstrinlinefullformat}{\glstrfullformat}%
8107     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
8108     \renewcommand*{\glstrinlinefullplformat}{\glstrfullplformat}%
8109     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

Reset `\glstrsubsequentfmt` etc in case a style changes this.

```
8110     \let\glstrsubsequentfmt\glstrdefaultsubsequentfmt
8111     \let\glstrsubsequentplfmt\glstrdefaultsubsequentplfmt
8112     \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
8113     \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
8114     #3}%
8115   }%
8116 }
```

abbreviationstyle

```
8117 \newcommand*{\renewabbreviationstyle}[3]{%
8118   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
8119   {%
8120     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
8121   }%
8122   {%
8123     \csdef{@glsabbrv@dispstyle@setup@#1}{%
```

Initialise hook to do nothing. The style may change this.

```
8124     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
8125     #2}%
8126     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
```

Assume in-line form is the same as first use. The style may change this.

```
8127     \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
8128     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
8129     \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
8130     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
8131     #3}%
8132   }%
8133 }
```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style’s name.

```
8134 \newcommand*{\letabbreviationstyle}[2]{%
8135   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
8136   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
8137 }
```

deprecated@abbrstyle

`\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```
8138 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
8139   \csdef{@glsabbrv@dispstyle@setup@#1}{%
8140     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
8141     \csuse{@glsabbrv@dispstyle@setup@#2}%
8142   }%
8143   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
8144 }
```

deprecatedAbbrStyle Generate warning for deprecated style use.

```
8145 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
8146   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
8147   use ‘#2’ instead}%
8148 }
```

eAbbrStyleSetup

```
8149 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
8150   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
8151   {%
8152     \PackageError{glossaries-extra}%
8153     {Unknown abbreviation style definitions ‘#1’}{}%
8154   }%
8155   {%
8156     \csname @glsabbrv@dispstyle@setup@#1\endcsname
8157   }%
8158 }
```

seAbbrStyleFmts

```
8159 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
8160   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
8161   {%
8162     \PackageError{glossaries-extra}%
8163     {Unknown abbreviation style formats ‘#1’}{}%
8164   }%
8165   {%
8166     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
8167   }%
8168 }
```

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, `\glsxtrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

xtrinsertinside Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```
8169 \newif\ifglsxtrinsertinside
8170 \glsxtrinsertinsidefalse
```

trlongshortname

```
8171 \newcommand*{\glsxtrlongshortname}{%
8172   \protect\glsabbrvfont{\the\glsshorttok}%
8173 }
```

long-short

```
8174 \newabbreviationstyle{long-short}%
8175 {%
```

Set accessibility attributes if enabled.

```
8176 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
8177 \renewcommand*{\CustomAbbreviationFields}{%
8178   name={\glxtrlongshortname},
8179   sort={\the\glsshorttok},
8180   first={\protect\glsfirstlongfont{\the\glslongtok}%
8181     \protect\glxtrfullsep{\the\glslabeltok}%
8182     \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
8183   firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
8184     \protect\glxtrfullsep{\the\glslabeltok}%
8185     \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%

8186   plural={\protect\glsabbrvfont{\the\glsshortpltok}}},%
8187   text={\protect\glsabbrvfont{\the\glsshorttok}}},%
8188   description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
8189 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8190   \glshasattribute{\the\glslabeltok}{regular}%
8191   {%
8192     \glissetattribute{\the\glslabeltok}{regular}{false}%
8193   }%
8194   {}%
8195 }%
8196 }%
8197 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
8198 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8199 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8200 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8201 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8202 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8203 \renewcommand*{\glxtrfullformat}[2]{%
8204   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8205   \ifglxtrininsertinside\else##2\fi
8206   \glxtrfullsep{##1}%
8207   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8208 }%
8209 \renewcommand*{\glxtrfullplformat}[2]{%
8210   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8211   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8212   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8213 }%
8214 \renewcommand*{\Glsxtrfullformat}[2]{%
8215   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8216   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
```

```

8217 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8218 }%
8219 \renewcommand*{\Glsxtrfullplformat}[2]{%
8220 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8221 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8222 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8223 }%
8224 }

```

Set this as the default style for general abbreviations:

```
8225 \setabbreviationstyle{long-short}
```

ngshortdescsort

```

8226 \newcommand*{\glsxtrlongshortdescsort}{%
8227 \expandonce\glsxtrorglong\space (\expandonce\glsxtrorgshort)%
8228 }

```

ngshortdescname

```

8229 \newcommand*{\glsxtrlongshortdescname}{%
8230 \protect\glslongfont{\the\glslongtok}
8231 \glsxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
8232 }

```

long-short-desc User supplies description. The long form is included in the name.

```

8233 \newabbreviationstyle{long-short-desc}%
8234 {%

```

Set accessibility attributes if enabled.

```
8235 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```

8236 \renewcommand*{\CustomAbbreviationFields}{%
8237 name={\glsxtrlongshortdescname},
8238 sort={\glsxtrlongshortdescsort},%
8239 first={\protect\glsfirstlongfont{\the\glslongtok}%
8240 \protect\glsxtrfullsep{\the\glslabeltok}%
8241 \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
8242 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
8243 \protect\glsxtrfullsep{\the\glslabeltok}%
8244 \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%

```

The text key should only have the short form.

```

8245 text={\protect\glsabbrvfont{\the\glsshorttok}},%
8246 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
8247 }%

```

Unset the regular attribute if it has been set.

```

8248 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8249 \glshasattribute{\the\glslabeltok}{regular}%

```



```

8250    {%
8251      \glsssetattribute{\the\glslabelltok}{regular}{false}%
8252    }%
8253    {}%
8254  }%
8255}%
8256{%
8257  \GlsXtrUseAbbrStyleFmts{long-short}%
8258}

```

trshortlongname

```

8259 \newcommand*{\glsxtrshortlongname}{%
8260   \protect\glsabbrvfont{\the\glsshorttok}%
8261 }

```

short-long Short form followed by long form in parenthesis on first use.

```

8262 \newabbreviationstyle{short-long}%
8263 {%

```

Set accessibility attributes if enabled.

```

8264   \glsxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

8265   \renewcommand*{\CustomAbbreviationFields}{%
8266     name={\glsxtrshortlongname},
8267     sort={\the\glsshorttok},
8268     description={\the\glslongtok},%
8269     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
8270       \protect\glsxtrfullsep{\the\glslabelltok}%
8271       \glsxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
8272     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
8273       \protect\glsxtrfullsep{\the\glslabelltok}%
8274       \glsxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
8275     text={\protect\glsabbrvfont{\the\glsshorttok}},%
8276     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}

```

Unset the regular attribute if it has been set.

```

8277   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8278     \glshasattribute{\the\glslabelltok}{regular}%
8279     {%
8280       \glsssetattribute{\the\glslabelltok}{regular}{false}%
8281     }%
8282     {}%
8283   }%
8284}%
8285{%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8286   \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
8287   \renewcommand*{\glsabbrvfont[1]}{\glsabbrvdefaultfont{##1}}%

```

```

8288 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8289 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8290 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8291 \renewcommand*{\glsxtrfullformat}[2]{%
8292   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8293   \ifglsxtrininsertinside\else##2\fi
8294   \glsxtrfullsep{##1}%
8295   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8296 }%
8297 \renewcommand*{\glsxtrfullplformat}[2]{%
8298   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8299   \ifglsxtrininsertinside\else##2\fi
8300   \glsxtrfullsep{##1}%
8301   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8302 }%
8303 \renewcommand*{\Glsxtrfullformat}[2]{%
8304   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8305   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8306   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8307 }%
8308 \renewcommand*{\Glsxtrfullplformat}[2]{%
8309   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8310   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8311   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8312 }%
8313 }

```

ortlongdescsort

```

8314 \newcommand*{\glsxtrshortlongdescsort}{\the\glsshorttok}

```

ortlongdescname

```

8315 \newcommand*{\glsxtrshortlongdescname}{%
8316   \protect\glsabbrvfont{\the\glsshorttok}
8317   \glsxtrparen{\protect\glslongfont{\the\glslongtok}}%
8318 }

```

short-long-desc User supplies description. The long form is included in the name.

```

8319 \newabbreviationstyle{short-long-desc}%
8320 {%

```

Set accessibility attributes if enabled.

```

8321 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

8322 \renewcommand*{\CustomAbbreviationFields}{%
8323   name={\glsxtrshortlongdescname},
8324   sort={\glsxtrshortlongdescsort},
8325   first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%

```

```

8326 \protect\glstrfullsep{\the\glslabeltok}%
8327 \glstrparen{\protect\glfirstlongfont{\the\glslongtok}}},%
8328 firstplural={\protect\glfirstabbrvfont{\the\glsshortpltok}%
8329 \protect\glstrfullsep{\the\glslabeltok}%
8330 \glstrparen{\protect\glfirstlongfont{\the\glslongpltok}}},%

8331 text={\protect\glabbrvfont{\the\glsshortttok}}},%

8332 plural={\protect\glabbrvfont{\the\glsshortpltok}}}%
8333 }%

Unset the regular attribute if it has been set.
8334 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8335 \glshasattribute{\the\glslabeltok}{regular}%
8336 {%
8337 \glsetattribute{\the\glslabeltok}{regular}{false}%
8338 }%
8339 {}%
8340 }%
8341 }%
8342 {%
8343 \GlsXtrUseAbbrStyleFmts{short-long}%
8344 }

```

`\glslongfootnotefont` Only used by the “footnote” styles.

```
8345 \newcommand*{\glslongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

`\glsshortfootnotefont` Only used by the “footnote” styles.

```
8346 \newcommand*{\glsshortfootnotefont}[1]{\glsshortfootnotefont{#1}}%
```

`\glstrabbrvfootnote`

`\glstrabbrvfootnote{<label>}{<long>}`

Command used by footnote abbreviation styles. The default definition ignores the first argument. The second argument *<long>* includes the font changing command and may be the singular or plural form, depending on the command that was used (for example, `\gls` or `\glspl`).

```
8347 \newcommand*{\glstrabbrvfootnote}[2]{\footnote{#2}}
```

`\glstrfootnotename`

```
8348 \newcommand*{\glstrfootnotename}{%
8349 \protect\glabbrvfont{\the\glsshortttok}%
8350 }
```

`\footnote` Short form followed by long form in footnote on first use.

```
8351 \newabbreviationstyle{footnote}%
8352 {%
```

Set accessibility attributes if enabled. (Add firstshortaccess since long form is hidden in a footnote on first use.)

```
8353 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
8354 \renewcommand*{\CustomAbbreviationFields}{%
8355   name={\glxtrfootnotename},
8356   sort={\the\glsshorttok},
8357   description={\the\glslongtok},%

8358   first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
8359     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8360     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8361   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
8362     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8363     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%

8364   text={\protect\glsabbrvfont{\the\glsshorttok}},%
8365   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.
```

```
8366 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8367   \glsssetAttribute{\the\glslabeltok}{nohyperfirst}{true}%
8368   \glshasattribute{\the\glslabeltok}{regular}%
8369   {%
8370     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
8371   }%
8372   {}%
8373 }%
8374 }%
8375 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
8376 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8377 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8378 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8379 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
8380 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
8381 \renewcommand*{\glxtrfullformat}[2]{%
8382   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8383   \ifglxtrininsertinside\else##2\fi
8384   \protect\glxtrabbrvfootnote{##1}%
8385   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8386 }%
8387 \renewcommand*{\glxtrfullplformat}[2]{%
8388   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8389   \ifglxtrininsertinside\else##2\fi
8390   \protect\glxtrabbrvfootnote{##1}%
```

```

8391     {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8392 }%
8393 \renewcommand*{\Glsxtrfullformat}[2]{%
8394   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8395   \ifglxtrinsertinside\else##2\fi
8396   \protect\glxtrabbrvfootnote{##1}%
8397   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8398 }%
8399 \renewcommand*{\Glsxtrfullplformat}[2]{%
8400   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8401   \ifglxtrinsertinside\else##2\fi
8402   \protect\glxtrabbrvfootnote{##1}%
8403   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8404 }%

```

The first use full form and the inline full form use the short (long) style.

```

8405 \renewcommand*{\glxtrinlinefullformat}[2]{%
8406   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8407   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8408   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8409 }%
8410 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8411   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8412   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8413   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8414 }%
8415 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8416   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8417   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8418   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8419 }%
8420 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8421   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8422   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8423   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8424 }%
8425 }

```

short-footnote

```

8426 \letabbreviationstyle{short-footnote}{footnote}

```

footnotedescname

```

8427 \newcommand*{\glxtrfootnotedescname}{%
8428   \protect\glsabbrvfont{\the\glsshorttok}%
8429   \protect\glxtrfullsep{\the\glslabeltok}%
8430   \protect\glxtrparen{\protect\glslongfont{\the\glslongtok}}}%
8431 }

```

footnotedescsort

```

8432 \newcommand*{\glxtrfootnotedescsort}{\the\glsshorttok}

```

t-footnote-desc Like short-footnote but with user supplied description.

```
8433 \newabbreviationstyle{short-footnote-desc}%  
8434 {%
```

Set accessibility attributes if enabled

```
8435 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```
8436 \renewcommand*{\CustomAbbreviationFields}{%  
8437   name={\glxtrfootnotedesname},  
8438   sort={\glxtrfootnotedesort},  
8439   first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%  
8440   \protect\glxtrabbrvfootnote{\the\glslabeltok}}%  
8441   {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%  
8442   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%  
8443   \protect\glxtrabbrvfootnote{\the\glslabeltok}}%  
8444   {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%  
8445   text={\protect\glsabrvfont{\the\glsshorttok}}},%  
8446   plural={\protect\glsabrvfont{\the\glsshortpltok}}}}%
```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
8447 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
8448   \glsetattribute{\the\glslabeltok}{nohyperfirst}{true}%  
8449   \glshasattribute{\the\glslabeltok}{regular}%  
8450   {%  
8451     \glsetattribute{\the\glslabeltok}{regular}{false}%  
8452   }%  
8453   {}%  
8454 }%  
8455 }%  
8456 {%  
8457   \GlsXtrUseAbbrStyleFmts{footnote}%  
8458 }
```

footnote-desc Synonym.

```
8459 \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.

```
8460 \newabbreviationstyle{postfootnote}%  
8461 {%
```

Set accessibility attributes if enabled. (Add firstshortaccess since long form is hidden in a footnote on first use.)

```
8462 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```
8463 \renewcommand*{\CustomAbbreviationFields}{%
```

```

8464     name={\glxtrfootnotename},
8465     sort={\the\glsshorttok},
8466     description={\the\glslongtok},%
8467     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
8468     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%

8469     text={\protect\glsabbrvfont{\the\glsshorttok}},%
8470     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.

```

8471 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8472   \csdef{glsxtrpostlink\glscategorylabel}{%
8473     \glxtrifwasfirstuse
8474     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8475       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}}%
8476       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%
8477   }%
8478   {}%
8479   }%
8480   \glshasattribute{\the\glslabeltok}{regular}%
8481   {%
8482     \glssetattribute{\the\glslabeltok}{regular}{false}%
8483     }%
8484     {}%
8485   }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

8486 \renewcommand*{\glxtrsetupfulldefs}{%
8487   \let\glxtrifwasfirstuse\@secondoftwo
8488   }%
8489 }%
8490 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8491 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8492 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8493 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8494 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
8495 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8496 \renewcommand*{\glxtrfullformat}[2]{%
8497   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8498   \ifglxtrininsertinside\else##2\fi
8499   }%
8500 \renewcommand*{\glxtrfullplformat}[2]{%

```

```

8501 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8502 \ifglxtrinsertinside\else##2\fi
8503 }%
8504 \renewcommand*{\Glsxtrfullformat}[2]{%
8505 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8506 \ifglxtrinsertinside\else##2\fi
8507 }%
8508 \renewcommand*{\Glsxtrfullplformat}[2]{%
8509 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8510 \ifglxtrinsertinside\else##2\fi
8511 }%

```

The first use full form and the inline full form use the short (long) style.

```

8512 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8513 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8514 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8515 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8516 }%
8517 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8518 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8519 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8520 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8521 }%
8522 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8523 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8524 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8525 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8526 }%
8527 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8528 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8529 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8530 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8531 }%
8532 }

```

rt-postfootnote

```

8533 \letabbreviationstyle{short-postfootnote}{postfootnote}

```

stfootnote-desc Like short-postfootnote but with user supplied description.

```

8534 \newabbreviationstyle{short-postfootnote-desc}%
8535 {}

```

Set accessibility attributes if enabled.

```

8536 \glsxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

8537 \renewcommand*{\CustomAbbreviationFields}{%
8538 name={\glsxtrfootnotedesname},
8539 sort={\glsxtrfootnotedesort},
8540 first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
8541 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%

```



```

8542   text={\protect\glsabbrvfont{\the\glsshorttok}},%
8543   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.

```

8544   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8545     \csdef{glsxtrpostlink\glscategorylabel}{%
8546       \glsxtrifwasfirstuse
8547       {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8548         \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}}%
8549         {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
8550     }%
8551   }%
8552 }%
8553 \glsattribute{\the\glslabeltok}{regular}%
8554 {%
8555   \glssetattribute{\the\glslabeltok}{regular}{false}%
8556 }%
8557 {}%
8558 }%

```

The footnote needs to be suppressed in the inline form, so `\glsxtrfull` must set the first use switch off.

```

8559   \renewcommand*{\glsxtrsetupfulldefs}{%
8560     \let\glsxtrifwasfirstuse\@secondoftwo
8561   }%
8562 }%
8563 {%
8564   \GlsXtrUseAbbrStyleFmts{postfootnote}%
8565 }

```

stfootnote-desc

```

8566 \letabbreviationstyle{postfootnote-desc}{short-postfootnote-desc}

```

shortnolongname

```

8567 \newcommand*{\glsxtrshortnolongname}{%
8568   \protect\glsabbrvfont{\the\glsshorttok}%
8569 }

```

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.

```

8570 \newabbreviationstyle{short}%
8571 {%

```

Set accessibility attributes if enabled.

```
8572 \glxstrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
8573 \renewcommand*{\CustomAbbreviationFields}{%
8574   name={\glxstrshortnolongname},
8575   sort={\the\glsshorttok},
8576   first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
8577   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
8578   text={\protect\glsabbrvfont{\the\glsshorttok}},
8579   plural={\protect\glsabbrvfont{\the\glsshortpltok}},
8580   description={\the\glslongtok}}%
8581 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8582   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8583 }%
8584 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
8585 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8586 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8587 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8588 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8589 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
8590 \renewcommand*{\glxtrinlinelinefullformat}[2]{%
8591   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
8592   \ifglxtrininsertinside##2\fi}%
8593   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8594   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8595 }%
8596 \renewcommand*{\glxtrinlinelinefullplformat}[2]{%
8597   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
8598   \ifglxtrininsertinside##2\fi}%
8599   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8600   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8601 }%
8602 \renewcommand*{\Glsxtrinlinelinefullformat}[2]{%
8603   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
8604   \ifglxtrininsertinside##2\fi}%
8605   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8606   \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
8607 }%
8608 \renewcommand*{\Glsxtrinlinelinefullplformat}[2]{%
8609   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
8610   \ifglxtrininsertinside##2\fi}%
8611   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8612   \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
8613 }%
```

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.

```

8614 \renewcommand*{\glxtrfullformat}[2]{%
8615   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8616   \ifglxtrinsertinside\else##2\fi
8617 }%
8618 \renewcommand*{\glxtrfullplformat}[2]{%
8619   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8620   \ifglxtrinsertinside\else##2\fi
8621 }%
8622 \renewcommand*{\Glsxtrfullformat}[2]{%
8623   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8624   \ifglxtrinsertinside\else##2\fi
8625 }%
8626 \renewcommand*{\Glsxtrfullplformat}[2]{%
8627   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8628   \ifglxtrinsertinside\else##2\fi
8629 }%
8630 }

```

Set this as the default style for acronyms:

```

8631 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

8632 \letabbreviationstyle{short-nolong}{short}

```

rt-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

8633 \newabbreviationstyle{short-nolong-noreg}%
8634 {%
8635   \GlsXtrUseAbbrStyleSetup{short-nolong}%
8636   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8637     \glshasattribute{\the\glslabeltok}{regular}%
8638     {%
8639       \glissetattribute{\the\glslabeltok}{regular}{false}%
8640     }%
8641   }%
8642 }%
8643 }%
8644 {%
8645   \GlsXtrUseAbbrStyleFmts{short-nolong}%
8646 }

```

trshortdescname

```

8647 \newcommand*{\glxtrshortdescname}{%
8648   \protect\glsabbrvfont{\the\glsshorttok}%
8649   \protect\glxtrfullsep{\the\glslabeltok}%
8650   \protect\glxtrparen{\protect\glslongfont{\the\glslongtok}}%
8651 }

```

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.

```
8652 \newabbreviationstyle{short-desc}%
8653 {%
```

Set accessibility attributes if enabled.

```
8654 \glxstrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
8655 \renewcommand*{\CustomAbbreviationFields}{%
8656   name={\glxstrshortdescname},
8657   sort={\the\glsshorttok},
8658   first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
8659   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
8660   text={\protect\glssabbrvfont{\the\glsshorttok}},
8661   plural={\protect\glssabbrvfont{\the\glsshortpltok}}}%
8662 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8663   \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
8664 }%
8665 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
8666 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8667 \renewcommand*{\glssabbrvfont[1]{\glssabbrvdefaultfont{##1}}}%
8668 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8669 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8670 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short format followed by the long form in parentheses.

```
8671 \renewcommand*{\glxtrinlinefullformat}[2]{%
8672   \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8673   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8674   \glxtrparen{\glsfirstlongfont{\glssaccesslong{##1}}}%
8675 }%
8676 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8677   \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8678   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8679   \glxtrparen{\glsfirstlongfont{\glssaccesslongpl{##1}}}%
8680 }%
8681 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8682   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8683   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8684   \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
8685 }%
8686 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8687   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8688   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8689   \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
8690 }%
```

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.

```

8691 \renewcommand*{\glxtrfullformat}[2]{%
8692   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8693   \ifglxtrinsertinside\else##2\fi
8694 }%
8695 \renewcommand*{\glxtrfullplformat}[2]{%
8696   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8697   \ifglxtrinsertinside\else##2\fi
8698 }%
8699 \renewcommand*{\Glsxtrfullformat}[2]{%
8700   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8701   \ifglxtrinsertinside\else##2\fi
8702 }%
8703 \renewcommand*{\Glsxtrfullplformat}[2]{%
8704   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8705   \ifglxtrinsertinside\else##2\fi
8706 }%
8707 }

```

ort-nolong-desc

```

8708 \letabbreviationstyle{short-nolong-desc}{short-desc}

```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

8709 \newabbreviationstyle{short-nolong-desc-noreg}%
8710 {%
8711   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%

```

Unset the regular attribute if it has been set.

```

8712 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8713   \glshasattribute{\the\glslabeltok}{regular}%
8714   {%
8715     \glsssetattribute{\the\glslabeltok}{regular}{false}%
8716   }%
8717   {}%
8718 }%
8719 }%
8720 {%
8721   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
8722 }

```

nolong-short Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

8723 \newabbreviationstyle{nolong-short}%
8724 {%
8725   \GlsXtrUseAbbrStyleSetup{short-nolong}%
8726 }%
8727 {%
8728   \GlsXtrUseAbbrStyleFmts{short-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8729 \renewcommand*{\glxtrinlinefullformat}[2]{%
8730   \protect\glsfirstlongfont{\glsaccesslong{##1}%
8731     \ifglxtrinsertinside##2\fi}%
8732   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8733   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8734 }%
8735 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8736   \protect\glsfirstlongfont{\glsaccesslongpl{##1}%
8737     \ifglxtrinsertinside##2\fi}%
8738   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8739   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8740 }%
8741 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8742   \protect\glsfirstlongfont{\glsaccesslong{##1}%
8743     \ifglxtrinsertinside##2\fi}%
8744   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8745   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
8746 }%
8747 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8748   \protect\glsfirstlongfont{\glsaccesslongpl{##1}%
8749     \ifglxtrinsertinside##2\fi}%
8750   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8751   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
8752 }%
8753 }

```

`ong-short-noreg` Like `nolong-short` but doesn't set the regular attribute.

```

8754 \newabbreviationstyle{nolong-short-noreg}%
8755 {%
8756   \GlsXtrUseAbbrStyleSetup{nolong-short}%

```

Unset the regular attribute if it has been set.

```

8757 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8758   \glshasattribute{\the\glslabeltok}{regular}%
8759   {%
8760     \glssetattribute{\the\glslabeltok}{regular}{false}%
8761   }%
8762   {}%
8763 }%
8764 }%
8765 {%
8766   \GlsXtrUseAbbrStyleFmts{nolong-short}%
8767 }

```

`noshortdescname`

```

8768 \newcommand*{\glxtrlongnoshortdescname}{%
8769   \protect\glslongfont{\the\glslongtok}%
8770 }

```

`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.

```

8771 \newabbreviationstyle{long-desc}%
8772 {%
8773   \renewcommand*{\CustomAbbreviationFields}{%
8774     name={\glxtrlongnoshortdescname},
8775     sort={\the\glslongtok},
8776     first={\protect\glslongfont{\the\glslongtok}},
8777     firstplural={\protect\glslongfont{\the\glslongpltok}},
8778     text={\glslongfont{\the\glslongtok}},
8779     plural={\glslongfont{\the\glslongpltok}}}%
8780   }%
8781   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8782     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
8783 }%
8784 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8785 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8786 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvdefaultfont{##1}}%
8787 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvdefaultfont{##1}}%
8788 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
8789 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8790 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
8791   \glslongfont{\glssaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8792   \ifglsxtrininsertinside \else##2\fi
8793 }%
8794 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
8795   \glslongfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8796   \ifglsxtrininsertinside \else##2\fi
8797 }%
8798 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8799   \glslongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8800   \ifglsxtrininsertinside \else##2\fi
8801 }%
8802 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8803   \glslongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8804   \ifglsxtrininsertinside \else##2\fi
8805 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8806 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8807   \glssfirstlongfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8808   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8809   \glsxtrparen{\protect\glssfirstabbrvfont{\glssaccessshort{##1}}}%
8810 }%

```

```

8811 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8812   \glsfirslongfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8813   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8814   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{##1}}}%
8815 }%
8816 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8817   \glsfirslongfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8818   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8819   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshort{##1}}}%
8820 }%
8821 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8822   \glsfirslongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8823   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8824   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{##1}}}%
8825 }%

```

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.

```

8826 \renewcommand*{\glxtrfullformat}[2]{%
8827   \glsfirslongfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8828   \ifglxtrininsertinside\else##2\fi
8829 }%
8830 \renewcommand*{\glxtrfullplformat}[2]{%
8831   \glsfirslongfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8832   \ifglxtrininsertinside\else##2\fi
8833 }%
8834 \renewcommand*{\Glsxtrfullformat}[2]{%
8835   \glsfirslongfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8836   \ifglxtrininsertinside\else##2\fi
8837 }%
8838 \renewcommand*{\Glsxtrfullplformat}[2]{%
8839   \glsfirslongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8840   \ifglxtrininsertinside\else##2\fi
8841 }%
8842 }

```

ng-noshort-desc Provide a synonym that matches similar styles.

```
8843 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

hort-desc-noreg Like long-noshort-desc but doesn't set the regular attribute.

```

8844 \newabbreviationstyle{long-noshort-desc-noreg}%
8845 {%
8846   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%

```

Unset the regular attribute if it has been set.

```

8847 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8848   \glshasattribute{\the\glslabeltok}{regular}%
8849   {%
8850     \glsssetattribute{\the\glslabeltok}{regular}{false}%
8851   }%

```



```

8852     {}%
8853   }%
8854 }%
8855 {%
8856   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
8857 }

```

longnoshortname

```

8858 \newcommand*{\glxtrlongnoshortname}{%
8859   \protect\glsabbrvfont{\the\glsshorttok}%
8860 }

```

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.

```

8861 \newabbreviationstyle{long}%
8862 {%

```

Set accessibility attributes if enabled.

```

8863   \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```

8864   \renewcommand*{\CustomAbbreviationFields}{%
8865     name={\glxtrlongnoshortname},
8866     sort={\the\glsshorttok},
8867     first={\protect\glsfirstlongfont{\the\glslongtok}},
8868     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
8869     text={\glslongfont{\the\glslongtok}},
8870     plural={\glslongfont{\the\glslongpltok}},%
8871     description={\the\glslongtok}%
8872   }%
8873   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8874     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
8875 }%
8876 {%
8877   \GlsXtrUseAbbrStyleFmts{long-desc}%
8878 }

```

long-noshort Provide a synonym that matches similar styles.

```

8879 \letabbreviationstyle{long-noshort}{long}

```

g-noshort-noreg Like long-noshort but doesn't set the regular attribute.

```

8880 \newabbreviationstyle{long-noshort-noreg}%
8881 {%
8882   \GlsXtrUseAbbrStyleSetup{long-noshort}%

```

Unset the regular attribute if it has been set.

```

8883   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8884     \glshassetAttribute{\the\glslabeltok}{regular}%
8885   }%

```

```

8886     \glsetattribute{\the\glslabeltok}{regular}{false}%
8887     }%
8888     {}%
8889     }%
8890 }%
8891 {%
8892   \GlsXtrUseAbbrStyleFmts{long-noshort}%
8893 }

```

1.7.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

`\glsxtrscfont` Maintained for backward-compatibility.

```
8894 \newcommand*{\glsxtrscfont}[1]{\textsc{#1}}
```

`\glsabbrvscfont` Added for consistent naming.

```
8895 \newcommand*{\glsabbrvscfont}{\glsxtrscfont}
```

`sxtrfirstscfont` Maintained for backward-compatibility.

```
8896 \newcommand*{\glsxtrfirstscfont}[1]{\glsabbrvscfont{#1}}
```

`irstabbrvscfont` Added for consistent naming.

```
8897 \newcommand*{\glfirstabbrvscfont}{\glsxtrfirstscfont}
```

and for the default short form suffix:

`\glsxtrscsuffix` `\protect` needs to come inside `\s` to avoid interfering with all caps.

```
8898 \newcommand*{\glsxtrscsuffix}{\protect\glstextup{\glxtrabbrvpluralsuffix}}
```

`long-short-sc`

```
8899 \newabbreviationstyle{long-short-sc}%
8900 {%

```

Set accessibility attributes if enabled.

```
8901   \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```

8902   \renewcommand*{\CustomAbbreviationFields}{%
8903     name={\glxtrlongshortname},
8904     sort={\the\glsshorttok},
8905     first={\protect\glfirstlongdefaultfont{\the\glslongtok}%
8906       \protect\glsxtrfullsep{\the\glslabeltok}%
8907       \glxtrparen{\protect\glfirstabbrvscfont{\the\glsshorttok}}},%
8908     firstplural={\protect\glfirstlongdefaultfont{\the\glslongpltok}%
8909       \protect\glsxtrfullsep{\the\glslabeltok}%
8910       \glxtrparen{\protect\glfirstabbrvscfont{\the\glsshortpltok}}},%
8911     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
8912     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},%

```

```

8913     description={\the\glslongtok}}%
8914 \renewcommand*\GlsXtrPostNewAbbreviation}{%
8915     \glsattribute{\the\glslabeltok}{regular}%
8916     {%
8917         \glssetattribute{\the\glslabeltok}{regular}{false}%
8918     }%
8919     {}%
8920 }%
8921 }%
8922 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

8923 \renewcommand*\abbrvpluralsuffix{\glsxtrscsuffix}%
8924 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
8925 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%

```

Use the default long fonts.

```

8926 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8927 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8928 \renewcommand*\glsxtrfullformat[2]{%
8929     \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8930     \ifglsxtrininsertinside\else##2\fi
8931     \glsxtrfullsep{##1}%
8932     \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
8933 }%
8934 \renewcommand*\glsxtrfullplformat[2]{%
8935     \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8936     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8937     \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
8938 }%
8939 \renewcommand*\Glsxtrfullformat[2]{%
8940     \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8941     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8942     \glsxtrparen{\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
8943 }%
8944 \renewcommand*\Glsxtrfullplformat[2]{%
8945     \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8946     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8947     \glsxtrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
8948 }%
8949 }

```

g-short-sc-desc

```

8950 \newabbreviationstyle{long-short-sc-desc}%
8951 {%

```

Set accessibility attributes if enabled.

```

8952 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```
8953 \renewcommand*{\CustomAbbreviationFields}{%
8954   name={\glxtrlongshortdescname},
8955   sort={\glxtrlongshortdescsort},%
8956   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8957     \protect\glxtrfullsep{\the\glslabeltok}%
8958     \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
8959   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8960     \protect\glxtrfullsep{\the\glslabeltok}%
8961     \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
8962   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
8963   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
8964 }
```

Unset the regular attribute if it has been set.

```
8965 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8966   \glshasattribute{\the\glslabeltok}{regular}%
8967   {%
8968     \glissetattribute{\the\glslabeltok}{regular}{false}%
8969   }%
8970   {}%
8971 }%
8972 }%
8973 {%
```

As long-short-sc style:

```
8974 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
8975 }
```

short-sc-long Now the short (long) version

```
8976 \newabbreviationstyle{short-sc-long}%
8977 {%
```

Set accessibility attributes if enabled.

```
8978 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
8979 \renewcommand*{\CustomAbbreviationFields}{%
8980   name={\glxtrshortlongname},
8981   sort={\the\glsshorttok},
8982   description={\the\glslongtok},%
8983   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
8984     \protect\glxtrfullsep{\the\glslabeltok}%
8985     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8986   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
8987     \protect\glxtrfullsep{\the\glslabeltok}%
8988     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8989   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
8990   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8991 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8992   \glshasattribute{\the\glslabelltok}{regular}}%
8993   {%
8994     \glissetattribute{\the\glslabelltok}{regular}{false}}%
8995   }%
8996   {}%
8997 }%
8998 }%
8999 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9000 \renewcommand*{\abbrvpluralsuffix}{\glsxtrscsuffix}%
9001 \renewcommand*{\glssabrvfont[1]{\glssabrvscfont{##1}}}%
9002 \renewcommand*{\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}}%
9003 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9004 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9005 \renewcommand*{\glsxtrfullformat}[2]{%
9006   \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9007   \ifglsxtrininsertinside\else##2\fi
9008   \glsxtrfullsep{##1}%
9009   \glsxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
9010 }%
9011 \renewcommand*{\glsxtrfullplformat}[2]{%
9012   \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9013   \ifglsxtrininsertinside\else##2\fi
9014   \glsxtrfullsep{##1}%
9015   \glsxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
9016 }%
9017 \renewcommand*{\Glsxtrfullformat}[2]{%
9018   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9019   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9020   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
9021 }%
9022 \renewcommand*{\Glsxtrfullplformat}[2]{%
9023   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9024   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9025   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
9026 }%
9027 }

```

rt-sc-long-desc As before but user provides description

```

9028 \newabbreviationstyle{short-sc-long-desc}%
9029 {%

```

Set accessibility attributes if enabled.

```

9030 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

9031 \renewcommand*{\CustomAbbreviationFields}{%

```

```

9032   name={\glxtrshortlongdescname},
9033   sort={\glxtrshortlongdescsort},
9034   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
9035     \protect\glxtrfullsep{\the\glslabeltok}%
9036     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
9037   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
9038     \protect\glxtrfullsep{\the\glslabeltok}%
9039     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
9040   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9041   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
9042 }%

```

Unset the regular attribute if it has been set.

```

9043 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9044   \glshasattribute{\the\glslabeltok}{regular}%
9045   {%
9046     \glissetattribute{\the\glslabeltok}{regular}{false}%
9047   }%
9048   {}%
9049 }%
9050 }%
9051 {%

```

As short-sc-long style:

```

9052 \GlsXtrUseAbbrStyleFmts{short-sc-long}%
9053 }

```

short-sc

```

9054 \newabbreviationstyle{short-sc}%
9055 {%

```

Set accessibility attributes if enabled.

```

9056 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9057 \renewcommand*{\CustomAbbreviationFields}{%
9058   name={\glxtrshortnolongname},
9059   sort={\the\glsshorttok},
9060   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
9061   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
9062   text={\protect\glsabbrvscfont{\the\glsshorttok}},
9063   plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
9064   description={\the\glslongtok}}%
9065 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9066   \glissetattribute{\the\glslabeltok}{regular}{true}}%
9067 }%
9068 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9069 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9070 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvscfont{##1}}%

```

```

9071 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
9072 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
9073 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

9074 \renewcommand*\glsxtrinlinefullformat}[2]{%
9075   \protect\glsfirstabbrvscfont{\glsaccesssshort{##1}}%
9076   \ifglsxtrininsertinside##2\fi}%
9077   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9078   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9079 }%
9080 \renewcommand*\glsxtrinlinefullplformat}[2]{%
9081   \protect\glsfirstabbrvscfont{\glsaccesssshortpl{##1}}%
9082   \ifglsxtrininsertinside##2\fi}%
9083   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9084   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9085 }%

9086 \renewcommand*\Glsxtrinlinefullformat}[2]{%
9087   \protect\glsfirstabbrvscfont{\Glsaccesssshort{##1}}%
9088   \ifglsxtrininsertinside##2\fi}%
9089   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9090   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9091 }%
9092 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
9093   \protect\glsfirstabbrvscfont{\Glsaccesssshortpl{##1}}%
9094   \ifglsxtrininsertinside##2\fi}%
9095   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9096   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9097 }%

```

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.

```

9098 \renewcommand*\glsxtrfullformat}[2]{%
9099   \glsfirstabbrvscfont{\glsaccesssshort{##1}}\ifglsxtrininsertinside##2\fi}%
9100   \ifglsxtrininsertinside\else##2\fi
9101 }%
9102 \renewcommand*\glsxtrfullplformat}[2]{%
9103   \glsfirstabbrvscfont{\glsaccesssshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9104   \ifglsxtrininsertinside\else##2\fi
9105 }%
9106 \renewcommand*\Glsxtrfullformat}[2]{%
9107   \glsfirstabbrvscfont{\glsaccesssshort{##1}}\ifglsxtrininsertinside##2\fi}%
9108   \ifglsxtrininsertinside\else##2\fi
9109 }%
9110 \renewcommand*\Glsxtrfullplformat}[2]{%
9111   \glsfirstabbrvscfont{\glsaccesssshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9112   \ifglsxtrininsertinside\else##2\fi
9113 }%
9114 }

```

short-sc-nolong

```
9115 \letabbreviationstyle{short-sc-nolong}{short-sc}
```

short-sc-desc

```
9116 \newabbreviationstyle{short-sc-desc}{%
```

```
9117 {%
```

Set accessibility attributes if enabled.

```
9118 \glxstrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```
9119 \renewcommand*{\CustomAbbreviationFields}{%
```

```
9120 name={\glxtrshortdescname},
```

```
9121 sort={\the\glsshorttok},
```

```
9122 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
```

```
9123 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
```

```
9124 text={\protect\glssabbrvscfont{\the\glsshorttok}},
```

```
9125 plural={\protect\glssabbrvscfont{\the\glsshortpltok}}}%
```

```
9126 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
9127 \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
```

```
9128 }%
```

```
9129 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
9130 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
```

```
9131 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvscfont{##1}}%
```

```
9132 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
```

```
9133 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
```

```
9134 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short format followed by the long form in parentheses.

```
9135 \renewcommand*{\glxtrinlinefullformat}[2]{%
```

```
9136 \glsfirstabbrvscfont{\glssaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
```

```
9137 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
```

```
9138 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}}%
```

```
9139 }%
```

```
9140 \renewcommand*{\glxtrinlinefullplformat}[2]{%
```

```
9141 \glsfirstabbrvscfont{\glssaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
```

```
9142 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
```

```
9143 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}}%
```

```
9144 }%
```

```
9145 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
```

```
9146 \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
```

```
9147 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
```

```
9148 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}}%
```

```
9149 }%
```

```
9150 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
```

```
9151 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
```

```
9152 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
```

```
9153 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}}%
```

```
9154 }%
```


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.

```

9155 \renewcommand*{\glxstrfullformat}[2]{%
9156   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9157   \ifglxtrinsertinside\else##2\fi
9158 }%
9159 \renewcommand*{\glxstrfullplformat}[2]{%
9160   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9161   \ifglxtrinsertinside\else##2\fi
9162 }%
9163 \renewcommand*{\Glsxtrfullformat}[2]{%
9164   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9165   \ifglxtrinsertinside\else##2\fi
9166 }%
9167 \renewcommand*{\Glsxtrfullplformat}[2]{%
9168   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9169   \ifglxtrinsertinside\else##2\fi
9170 }%
9171 }

```

-sc-nolong-desc

```

9172 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

```

nolong-short-sc

```

9173 \newabbreviationstyle{nolong-short-sc}%
9174 {%
9175   \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%
9176 }%
9177 {%
9178   \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

9179 \renewcommand*{\glxstrinlinefullformat}[2]{%
9180   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
9181   \ifglxtrinsertinside##2\fi}%
9182   \ifglxtrinsertinside\else##2\fi\glxstrfullsep{##1}%
9183   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9184 }%
9185 \renewcommand*{\glxstrinlinefullplformat}[2]{%
9186   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
9187   \ifglxtrinsertinside##2\fi}%
9188   \ifglxtrinsertinside\else##2\fi\glxstrfullsep{##1}%
9189   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9190 }%
9191 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9192   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
9193   \ifglxtrinsertinside##2\fi}%
9194   \ifglxtrinsertinside\else##2\fi\glxstrfullsep{##1}%
9195   \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%

```

```

9196 }%
9197 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9198   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
9199     \ifglxtrinsertinside##2\fi}%
9200   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9201   \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
9202 }%
9203 }

```

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.

```

9204 \newabbreviationstyle{long-noshort-sc}%
9205 {%
9206   \renewcommand*{\CustomAbbreviationFields}{%
9207     name={\glxtrlongnoshortname},
9208     sort={\the\glsshorttok},
9209     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
9210     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
9211     text={\protect\glslongdefaultfont{\the\glslongtok}},
9212     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
9213     description={\the\glslongtok}%
9214   }%
9215   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9216     \glsetattribute{\the\glslabeltok}{regular}{true}}%
9217 }%
9218 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9219 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9220 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9221 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9222 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9223 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9224 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9225   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9226   \ifglxtrinsertinside \else##2\fi
9227 }%
9228 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9229   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9230   \ifglxtrinsertinside \else##2\fi
9231 }%
9232 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9233   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9234   \ifglxtrinsertinside \else##2\fi
9235 }%
9236 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9237   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9238   \ifglxtrinsertinside \else##2\fi

```

9239 }%

The inline full form displays the long format followed by the short form in parentheses.

```
9240 \renewcommand*{\glxstrinlinefullformat}[2]{%
9241   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxstrinsertinside##2\fi}%
9242   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9243   \glxstrparen{\protect\glsfirstabbrvscfont{\glssaccessshort{##1}}}%
9244 }%
9245 \renewcommand*{\glxstrinlinefullplformat}[2]{%
9246   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxstrinsertinside##2\fi}%
9247   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9248   \glxstrparen{\protect\glsfirstabbrvscfont{\glssaccessshortpl{##1}}}%
9249 }%
9250 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9251   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxstrinsertinside##2\fi}%
9252   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9253   \glxstrparen{\protect\glsfirstabbrvscfont{\glssaccessshort{##1}}}%
9254 }%
9255 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9256   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxstrinsertinside##2\fi}%
9257   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9258   \glxstrparen{\protect\glsfirstabbrvscfont{\glssaccessshortpl{##1}}}%
9259 }%
```

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.

```
9260 \renewcommand*{\glxstrfullformat}[2]{%
9261   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxstrinsertinside##2\fi}%
9262   \ifglxstrinsertinside\else##2\fi
9263 }%
9264 \renewcommand*{\glxstrfullplformat}[2]{%
9265   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxstrinsertinside##2\fi}%
9266   \ifglxstrinsertinside\else##2\fi
9267 }%
9268 \renewcommand*{\Glsxtrfullformat}[2]{%
9269   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxstrinsertinside##2\fi}%
9270   \ifglxstrinsertinside\else##2\fi
9271 }%
9272 \renewcommand*{\Glsxtrfullplformat}[2]{%
9273   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxstrinsertinside##2\fi}%
9274   \ifglxstrinsertinside\else##2\fi
9275 }%
9276 }
```

long-sc Backward compatibility:

```
9277 \@glxstr@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 \glssshort.

```
9278 \newabbreviationstyle{long-noshort-sc-desc}%
```

```

9279 {%
9280   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9281 }%
9282 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9283   \renewcommand*\abbrvpluralsuffix{\glxtrscsuffix}%
9284   \renewcommand*\glssabbrvfont[1]{\glssabbrvscfont{##1}}%
9285   \renewcommand*\glssfirstabbrvfont[1]{\glssfirstabbrvscfont{##1}}%
9286   \renewcommand*\glssfirstlongfont[1]{\glssfirstlongdefaultfont{##1}}%
9287   \renewcommand*\glsslongfont[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9288   \renewcommand*\glxtrsubsequentfmt}[2]{%
9289     \glsslongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9290     \ifglxtrininsertinside \else##2\fi
9291   }%
9292   \renewcommand*\glxtrsubsequentplfmt}[2]{%
9293     \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9294     \ifglxtrininsertinside \else##2\fi
9295   }%
9296   \renewcommand*\Glsxtrsubsequentfmt}[2]{%
9297     \glsslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9298     \ifglxtrininsertinside \else##2\fi
9299   }%
9300   \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
9301     \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9302     \ifglxtrininsertinside \else##2\fi
9303   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9304   \renewcommand*\glxtrininlinefullformat}[2]{%
9305     \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9306     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9307     \glxtrparen{\protect\glssfirstabbrvscfont{\glssaccessshort{##1}}}%
9308   }%
9309   \renewcommand*\glxtrininlinefullplformat}[2]{%
9310     \glssfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9311     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9312     \glxtrparen{\protect\glssfirstabbrvscfont{\glssaccessshortpl{##1}}}%
9313   }%
9314   \renewcommand*\Glsxtrininlinefullformat}[2]{%
9315     \glssfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9316     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9317     \glxtrparen{\protect\glssfirstabbrvscfont{\glssaccessshort{##1}}}%
9318   }%
9319   \renewcommand*\Glsxtrininlinefullplformat}[2]{%
9320     \glssfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9321     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9322     \glxtrparen{\protect\glssfirstabbrvscfont{\glssaccessshortpl{##1}}}%
9323   }%

```

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.

```

9324 \renewcommand*{\glxtrfullformat}[2]{%
9325   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9326   \ifglxtrinsertinside\else##2\fi
9327 }%
9328 \renewcommand*{\glxtrfullplformat}[2]{%
9329   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9330   \ifglxtrinsertinside\else##2\fi
9331 }%
9332 \renewcommand*{\Glsxtrfullformat}[2]{%
9333   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9334   \ifglxtrinsertinside\else##2\fi
9335 }%
9336 \renewcommand*{\Glsxtrfullplformat}[2]{%
9337   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9338   \ifglxtrinsertinside\else##2\fi
9339 }%
9340 }

```

long-desc-sc Backward compatibility:

```

9341 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

9342 \newabbreviationstyle{short-sc-footnote}%
9343 {%

```

Set accessibility attributes if enabled.

```

9344 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9345 \renewcommand*{\CustomAbbreviationFields}{%
9346   name={\glxtrfootnotename},
9347   sort={\the\glsshorttok},
9348   description={\the\glslongtok},%
9349   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
9350     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9351     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9352   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
9353     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9354     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9355   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9356   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.

```

9357 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9358   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9359   \glsattribute{\the\glslabeltok}{regular}%
9360 }%

```

```

9361      \glsetattribute{\the\glslabeltok}{regular}{false}%
9362    }%
9363    {}%
9364  }%
9365}%
9366{%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9367  \renewcommand*\abbrvpluralsuffix{\glxtrscsuffix}%
9368  \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
9369  \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
9370  \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
9371  \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9372  \renewcommand*\glsxtrfullformat}[2]{%
9373    \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9374    \ifglsxtrininsertinside\else##2\fi
9375    \protect\glsxtrabbrvfootnote{##1}%
9376    {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9377  }%
9378  \renewcommand*\glsxtrfullplformat}[2]{%
9379    \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9380    \ifglsxtrininsertinside\else##2\fi
9381    \protect\glsxtrabbrvfootnote{##1}%
9382    {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9383  }%
9384  \renewcommand*\Glsxtrfullformat}[2]{%
9385    \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9386    \ifglsxtrininsertinside\else##2\fi
9387    \protect\glsxtrabbrvfootnote{##1}%
9388    {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
9389  }%
9390  \renewcommand*\Glsxtrfullplformat}[2]{%
9391    \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9392    \ifglsxtrininsertinside\else##2\fi
9393    \protect\glsxtrabbrvfootnote{##1}%
9394    {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
9395  }%

```

The first use full form and the inline full form use the short (long) style.

```

9396  \renewcommand*\glsxtrininlinefullformat}[2]{%
9397    \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9398    \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9399    \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9400  }%
9401  \renewcommand*\glsxtrininlinefullplformat}[2]{%
9402    \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9403    \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9404    \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9405  }%

```

```

9406 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9407   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9408   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9409   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
9410 }%
9411 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9412   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9413   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9414   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
9415 }%
9416 }

```

footnote-sc Backward compatibility:

```

9417 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}

```

c-footnote-desc Like short-sc-footnote but with user supplied description.

```

9418 \newabbreviationstyle{short-sc-footnote-desc}%
9419 {%

```

Set accessibility attributes if enabled.

```

9420 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

9421 \renewcommand*{\CustomAbbreviationFields}{%
9422   name={\glxtrfootnotedesname},
9423   sort={\glxtrfootnotedesort},
9424   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
9425     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9426     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9427   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
9428     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9429     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9430   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9431   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.

```

9432 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9433   \glsetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9434   \glshasattribute{\the\glslabeltok}{regular}%
9435   {%
9436     \glsetattribute{\the\glslabeltok}{regular}{false}%
9437   }%
9438   {}%
9439 }%
9440 }%
9441 {%
9442   \GlsXtrUseAbbrStyleFmts{short-sc-footnote}%
9443 }

```

sc-postfootnote

```
9444 \newabbreviationstyle{short-sc-postfootnote}%
9445 {%
```

Set accessibility attributes if enabled.

```
9446 \glxstrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
9447 \renewcommand*{\CustomAbbreviationFields}{%
9448   name={\glxstrfootnotename},
9449   sort={\the\glsshorttok},
9450   description={\the\glslongtok},%
9451   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
9452   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
9453   text={\protect\glssabbrvscfont{\the\glsshorttok}},%
9454   plural={\protect\glssabbrvscfont{\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.

```
9455 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9456   \csdef{glxstrpostlink\glscategorylabel}{%
9457     \glxstrifwasfirstuse
9458     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
9459       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9460       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
9461     }%
9462   }%
9463 }%
9464 \glshasattribute{\the\glslabeltok}{regular}%
9465 {%
9466   \glissetattribute{\the\glslabeltok}{regular}{false}%
9467 }%
9468 {}%
9469 }%
```

The footnote needs to be suppressed in the inline form, so \glxstrfull must set the first use switch off.

```
9470 \renewcommand*{\glxstrsetupfulldefs}{%
9471   \let\glxstrifwasfirstuse\@secondoftwo
9472 }%
9473 }%
9474 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
9475 \renewcommand*{\abbrvpluralsuffix}{\glxstrscsuffix}%
9476 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvscfont{##1}}%
9477 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9478 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9479 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```


The full format displays the short form. The long form is deferred.

```

9480 \renewcommand*{\glxtrfullformat}[2]{%
9481   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9482   \ifglxtrinsertinside\else##2\fi
9483 }%
9484 \renewcommand*{\glxtrfullplformat}[2]{%
9485   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9486   \ifglxtrinsertinside\else##2\fi
9487 }%
9488 \renewcommand*{\Glsxtrfullformat}[2]{%
9489   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9490   \ifglxtrinsertinside\else##2\fi
9491 }%
9492 \renewcommand*{\Glsxtrfullplformat}[2]{%
9493   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9494   \ifglxtrinsertinside\else##2\fi
9495 }%

```

The first use full form and the inline full form use the short (long) style.

```

9496 \renewcommand*{\glxtrinlinefullformat}[2]{%
9497   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9498   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9499   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9500 }%
9501 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9502   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9503   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9504   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9505 }%
9506 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9507   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9508   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9509   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9510 }%
9511 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9512   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9513   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9514   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9515 }%
9516 }

```

postfootnote-sc Backward compatibility:

```

9517 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}

```

stfootnote-desc Like short-sc-footnote but with user supplied description.

```

9518 \newabbreviationstyle{short-sc-postfootnote-desc}%
9519 {}

```

Set accessibility attributes if enabled.

```

9520 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

9521 \renewcommand*{\CustomAbbreviationFields}{%
9522     name={\glxtrfootnotedescname},
9523     sort={\glxtrfootnotedescsort},
9524     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
9525     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
9526     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9527     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.

```

9528 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9529     \csdef{glxtrpostlink\glscategorylabel}{%
9530         \glxtrifwasfirstuse
9531         {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

9532             \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9533             {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}}%
9534         }%
9535     }%
9536 }%
9537 \glshasattribute{\the\glslabeltok}{regular}%
9538 {%
9539     \glissetattribute{\the\glslabeltok}{regular}{false}%
9540 }%
9541 {}%
9542 }%

```

The footnote needs to be suppressed in the inline form, so \glxtrfull must set the first use switch off.

```

9543 \renewcommand*{\glxtrsetupfulldefs}{%
9544     \let\glxtrifwasfirstuse\@secondoftwo
9545 }%
9546 }%
9547 {%
9548     \GlsXtrUseAbbrStyleFmts{short-sc-postfootnote}%
9549 }

```

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.

```

9550 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}

```

\glsabbrvsmfont Added for consistent naming.

```

9551 \newcommand*{\glsabbrvsmfont}{\glxtrsmfont}

```

sxtrfirstsmfont Maintained for backward compatibility.

```
9552 \newcommand*{\glxsxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}
```

irstabbrvsmfont Added for consistent naming.

```
9553 \newcommand*{\glsfirstabbrvsmfont}{\glxsxtrfirstsmfont}
```

and for the default short form suffix:

\glsxtrsmsuffix

```
9554 \newcommand*{\glsxtrsmsuffix}{\glsxtrabbrvpluralsuffix}
```

long-short-sm

```
9555 \newabbreviationstyle{long-short-sm}%
```

```
9556 {%
```

Set accessibility attributes if enabled.

```
9557 \glxsxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
9558 \renewcommand*{\CustomAbbreviationFields}{%
```

```
9559   name={\glsxtrlongshortname},
```

```
9560   sort={\the\glsshorttok},
```

```
9561   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
```

```
9562     \protect\glxsxtrfullsep{\the\glslabeltok}%
```

```
9563     \glxsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
```

```
9564   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
```

```
9565     \protect\glxsxtrfullsep{\the\glslabeltok}%
```

```
9566     \glxsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
```

```
9567   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
```

```
9568   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%
```

```
9569   description={\the\glslongtok}}%
```

```
9570 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
9571   \glshasattribute{\the\glslabeltok}{regular}%
```

```
9572   {%
```

```
9573     \glissetattribute{\the\glslabeltok}{regular}{false}%
```

```
9574   }%
```

```
9575   }%
```

```
9576 }%
```

```
9577 }%
```

```
9578 {%
```

```
9579 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
```

```
9580 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
```

```
9581 \renewcommand*\abbrvpluralsuffix{\glsxtrsmsuffix}%
```

Use the default long fonts.

```
9582 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
```

```
9583 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline form are the same for this style.

```

9584 \renewcommand*{\glxtrfullformat}[2]{%
9585   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9586   \ifglxtrinsertinside\else##2\fi
9587   \glxtrfullsep{##1}%
9588   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
9589 }%
9590 \renewcommand*{\glxtrfullplformat}[2]{%
9591   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9592   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9593   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
9594 }%
9595 \renewcommand*{\Glsxtrfullformat}[2]{%
9596   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9597   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9598   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
9599 }%
9600 \renewcommand*{\Glsxtrfullplformat}[2]{%
9601   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9602   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9603   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
9604 }%
9605 }

```

g-short-sm-desc

```

9606 \newabbreviationstyle{long-short-sm-desc}%
9607 {%

```

Set accessibility attributes if enabled.

```

9608 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

9609 \renewcommand*{\CustomAbbreviationFields}{%
9610   name={\glxtrlongshortdescname},
9611   sort={\glxtrlongshortdescsort},%
9612   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
9613     \protect\glxtrfullsep{\the\glslabeltok}%
9614     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
9615   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
9616     \protect\glxtrfullsep{\the\glslabeltok}%
9617     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
9618   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
9619   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
9620 }%

```

Unset the regular attribute if it has been set.

```

9621 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9622   \glshasattribute{\the\glslabeltok}{regular}%
9623   {%
9624     \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

9625 }%
9626 {}%
9627 }%
9628 }%
9629 {%

```

As long-short-sm style:

```

9630 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
9631 }

```

short-sm-long Now the short (long) version

```

9632 \newabbreviationstyle{short-sm-long}%
9633 {%

```

Set accessibility attributes if enabled.

```

9634 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

9635 \renewcommand*{\CustomAbbreviationFields}{%
9636   name={\glxtrshortlongname},
9637   sort={\the\glsshorttok},
9638   description={\the\glslongtok},%
9639   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
9640     \protect\glxtrfullsep{\the\glslabeltok}%
9641     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
9642   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
9643     \protect\glxtrfullsep{\the\glslabeltok}%
9644     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
9645   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
9646   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

9647 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9648   \glshasattribute{\the\glslabeltok}{regular}%
9649   {%
9650     \glissetattribute{\the\glslabeltok}{regular}{false}%
9651   }%
9652   {}%
9653 }%
9654 }%
9655 {%

```

```

9656 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
9657 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
9658 \renewcommand*\{abbrvpluralsuffix}{\glxtrsmsuffix}%
9659 \renewcommand*\{glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9660 \renewcommand*\{glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9661 \renewcommand*{\glxtrfullformat}[2]{%
9662   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%

```

```

9663 \ifglxtrinsertinside\else##2\fi
9664 \glxtrfullsep{##1}%
9665 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
9666 }%
9667 \renewcommand*{\glxtrfullplformat}[2]{%
9668 \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9669 \ifglxtrinsertinside\else##2\fi
9670 \glxtrfullsep{##1}%
9671 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
9672 }%
9673 \renewcommand*{\Glsxtrfullformat}[2]{%
9674 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9675 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9676 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
9677 }%
9678 \renewcommand*{\Glsxtrfullplformat}[2]{%
9679 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9680 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9681 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
9682 }%
9683 }

```

rt-sm-long-desc As before but user provides description

```

9684 \newabbreviationstyle{short-sm-long-desc}%
9685 {}%

```

Set accessibility attributes if enabled.

```

9686 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

9687 \renewcommand*{\CustomAbbreviationFields}{%
9688 name={\glxtrshortlongdescname},
9689 sort={\glxtrshortlongdescsort},
9690 first={\protect\glsfirstabbrvsmfont{\the\glssshorttok}}%
9691 \protect\glxtrfullsep{\the\glslabeltok}}%
9692 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
9693 firstplural={\protect\glsfirstabbrvsmfont{\the\glssshortpltok}}%
9694 \protect\glxtrfullsep{\the\glslabeltok}}%
9695 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
9696 text={\protect\glsabbrvsmfont{\the\glssshorttok}},%
9697 plural={\protect\glsabbrvsmfont{\the\glssshortpltok}}}%
9698 }%

```

Unset the regular attribute if it has been set.

```

9699 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9700 \glshasattribute{\the\glslabeltok}{regular}%
9701 {}%
9702 \glissetattribute{\the\glslabeltok}{regular}{false}%
9703 }%
9704 {}%

```

```

9705 }%
9706 }%
9707 {%

```

As short-sm-long style:

```

9708 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
9709 }

```

short-sm

```

9710 \newabbreviationstyle{short-sm}%
9711 {%

```

Set accessibility attributes if enabled.

```

9712 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

9713 \renewcommand*\CustomAbbreviationFields{%
9714   name={\glxtrshortnolongname},
9715   sort={\the\glsshorttok},
9716   first={\protect\glfirstabbrvsmfont{\the\glsshorttok}},
9717   firstplural={\protect\glfirstabbrvsmfont{\the\glsshortpltok}},
9718   text={\protect\glabbrvsmfont{\the\glsshorttok}},
9719   plural={\protect\glabbrvsmfont{\the\glsshortpltok}},
9720   description={\the\glslongtok}}%
9721 \renewcommand*\GlsXtrPostNewAbbreviation{%
9722   \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9723 }%
9724 {%

9725 \renewcommand*\glabbrvfont[1]{\glabbrvsmfont{##1}}%
9726 \renewcommand*\glfirstabbrvfont[1]{\glfirstabbrvsmfont{##1}}%
9727 \renewcommand*\abbrvpluralsuffix{\glxtrsmsuffix}%
9728 \renewcommand*\glfirstlongfont[1]{\glfirstlongdefaultfont{##1}}%
9729 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

9730 \renewcommand*\glxtrinlinefullformat[2]{%
9731   \protect\glfirstabbrvsmfont{\glsaccessshort{##1}}%
9732   \ifglxtrininsertinside##2\fi}%
9733 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9734 \glxtrparen{\glfirstlongdefaultfont{\glsaccesslong{##1}}}%
9735 }%

9736 \renewcommand*\glxtrinlinefullplformat[2]{%
9737   \protect\glfirstabbrvsmfont{\glsaccessshortpl{##1}}%
9738   \ifglxtrininsertinside##2\fi}%
9739 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9740 \glxtrparen{\glfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9741 }%

9742 \renewcommand*\Glsxtrinlinefullformat[2]{%
9743   \protect\glfirstabbrvsmfont{\Glsaccessshort{##1}}%

```

```

9744     \ifglxtrinsertinside##2\fi}%
9745     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9746     \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
9747 }%
9748 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9749     \protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}%
9750         \ifglxtrinsertinside##2\fi}%
9751     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9752     \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
9753 }%

```

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.

```

9754 \renewcommand*{\glxtrfullformat}[2]{%
9755     \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9756     \ifglxtrinsertinside\else##2\fi
9757 }%
9758 \renewcommand*{\glxtrfullplformat}[2]{%
9759     \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9760     \ifglxtrinsertinside\else##2\fi
9761 }%
9762 \renewcommand*{\Glsxtrfullformat}[2]{%
9763     \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9764     \ifglxtrinsertinside\else##2\fi
9765 }%
9766 \renewcommand*{\Glsxtrfullplformat}[2]{%
9767     \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9768     \ifglxtrinsertinside\else##2\fi
9769 }%
9770 }

```

short-sm-nolong

```

9771 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

9772 \newabbreviationstyle{short-sm-desc}%
9773 {}

```

Set accessibility attributes if enabled.

```

9774 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9775 \renewcommand*{\CustomAbbreviationFields}{%
9776     name={\glxtrshortdescname},
9777     sort={\the\glssshorttok},
9778     first={\protect\glsfirstabbrvsmfont{\the\glssshorttok}},
9779     firstplural={\protect\glsfirstabbrvsmfont{\the\glssshortpltok}},
9780     text={\protect\glssabbrvsmfont{\the\glssshorttok}},
9781     plural={\protect\glssabbrvsmfont{\the\glssshortpltok}}}%
9782 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```



```

9783 \glsssetAttribute{\the\glslabelltok}{regular}{true}}%
9784 }%
9785 {%

```

```

9786 \renewcommand*\glssabrvfont[1]{\glssabrvsmfont{##1}}%
9787 \renewcommand*\glssfirstabrvfont[1]{\glssfirstabrvsmfont{##1}}%
9788 \renewcommand*\glssabrvpluralsuffix{\glssxtrmsuffix}%
9789 \renewcommand*\glssfirstlongfont[1]{\glssfirstlongdefaultfont{##1}}%
9790 \renewcommand*\glsslongfont[1]{\glsslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

9791 \renewcommand*\glssxtrinlinefullformat}[2]{%
9792 \glssfirstabrvsmfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
9793 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9794 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslong{##1}}}%
9795 }%
9796 \renewcommand*\glssxtrinlinefullplformat}[2]{%
9797 \glssfirstabrvsmfont{\glssaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
9798 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9799 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
9800 }%
9801 \renewcommand*\Glsxtrinlinefullformat}[2]{%
9802 \glssfirstabrvsmfont{\Glsaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
9803 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9804 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslong{##1}}}%
9805 }%
9806 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
9807 \glssfirstabrvsmfont{\Glsaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
9808 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9809 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
9810 }%

```

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.

```

9811 \renewcommand*\glssxtrfullformat}[2]{%
9812 \glssfirstabrvsmfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
9813 \ifglssxtrininsertinside\else##2\fi
9814 }%
9815 \renewcommand*\glssxtrfullplformat}[2]{%
9816 \glssfirstabrvsmfont{\glssaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
9817 \ifglssxtrininsertinside\else##2\fi
9818 }%
9819 \renewcommand*\Glsxtrfullformat}[2]{%
9820 \glssfirstabrvsmfont{\Glsaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
9821 \ifglssxtrininsertinside\else##2\fi
9822 }%
9823 \renewcommand*\Glsxtrfullplformat}[2]{%
9824 \glssfirstabrvsmfont{\Glsaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
9825 \ifglssxtrininsertinside\else##2\fi
9826 }%

```

9827 }

-sm-nolong-desc

9828 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}

nolong-short-sm

9829 \newabbreviationstyle{nolong-short-sm}{%

9830 {%

9831 \GlsXtrUseAbbrStyleSetup{short-sm-nolong}{%

9832 }%

9833 {%

9834 \GlsXtrUseAbbrStyleFmts{short-sm-nolong}{%

The inline full form displays the long form followed by the short form in parentheses.

9835 \renewcommand*{\glxtrinlinefullformat}[2]{%

9836 \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}}%

9837 \ifglxtrininsertinside##2\fi}%

9838 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%

9839 \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}}%

9840 }%

9841 \renewcommand*{\glxtrinlinefullplformat}[2]{%

9842 \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}%

9843 \ifglxtrininsertinside##2\fi}%

9844 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%

9845 \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}}%

9846 }%

9847 \renewcommand*{\Glsxtrinlinefullformat}[2]{%

9848 \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}}%

9849 \ifglxtrininsertinside##2\fi}%

9850 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%

9851 \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}}%

9852 }%

9853 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%

9854 \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}%

9855 \ifglxtrininsertinside##2\fi}%

9856 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%

9857 \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}}%

9858 }%

9859 }

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like \glsshort.

9860 \newabbreviationstyle{long-noshort-sm}{%

9861 {%

Set accessibility attributes if enabled.

9862 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

Setup the default fields.

9863 \renewcommand*{\CustomAbbreviationFields}{%

```

9864     name={\glxtrlongnosshortname},
9865     sort={\the\glssshorttok},
9866     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
9867     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
9868     text={\protect\glslongdefaultfont{\the\glslongtok}},
9869     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
9870     description={\the\glslongtok}%
9871 }%
9872 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9873   \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9874 }%
9875 {%

9876 \renewcommand*{\glabbrvfont}[1]{\glabbrvsmfont{##1}}%
9877 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
9878 \renewcommand*{\abbrvpluralsuffix}{\glxtrsmsuffix}%
9879 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9880 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9881 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9882   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9883   \ifglxtrininsertinside \else##2\fi
9884 }%
9885 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9886   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9887   \ifglxtrininsertinside \else##2\fi
9888 }%
9889 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9890   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9891   \ifglxtrininsertinside \else##2\fi
9892 }%
9893 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9894   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9895   \ifglxtrininsertinside \else##2\fi
9896 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9897 \renewcommand*{\glxtrininlinefullformat}[2]{%
9898   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9899   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9900   \glxtrparen{\protect\glsfirstabbrvsmfont{\glssaccessshort{##1}}}%
9901 }%
9902 \renewcommand*{\glxtrininlinefullplformat}[2]{%
9903   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9904   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9905   \glxtrparen{\protect\glsfirstabbrvsmfont{\glssaccessshortpl{##1}}}%
9906 }%
9907 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
9908   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%

```

```

9909     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9910     \glxtrparen{\protect\glsfirstabbrvsmfont{\glssaccessshort{##1}}}%
9911 }%
9912 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9913     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9914     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9915     \glxtrparen{\protect\glsfirstabbrvsmfont{\glssaccessshortpl{##1}}}%
9916 }%

```

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.

```

9917 \renewcommand*{\glxtrfullformat}[2]{%
9918     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9919     \ifglxtrinsertinside\else##2\fi
9920 }%
9921 \renewcommand*{\glxtrfullplformat}[2]{%
9922     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9923     \ifglxtrinsertinside\else##2\fi
9924 }%
9925 \renewcommand*{\Glsxtrfullformat}[2]{%
9926     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9927     \ifglxtrinsertinside\else##2\fi
9928 }%
9929 \renewcommand*{\Glsxtrfullplformat}[2]{%
9930     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9931     \ifglxtrinsertinside\else##2\fi
9932 }%
9933 }

```

long-sm Backward compatibility:

```

9934 \@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`.

```

9935 \newabbreviationstyle{long-noshort-sm-desc}%
9936 {%
9937     \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9938 }%
9939 {%
9940     \renewcommand*{\glssabbrvfont}[1]{\glssabbrvsmfont{##1}}%
9941     \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
9942     \renewcommand*{\abbrvpluralsuffix}{\glsxtrsmsuffix}%
9943     \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9944     \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9945 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9946     \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9947     \ifglxtrinsertinside \else##2\fi

```

```

9948 }%
9949 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9950   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9951   \ifglxtrinsertinside \else##2\fi
9952 }%
9953 \renewcommand*{\Glsxtrsubsequentffmt}[2]{%
9954   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9955   \ifglxtrinsertinside \else##2\fi
9956 }%
9957 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9958   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9959   \ifglxtrinsertinside \else##2\fi
9960 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9961 \renewcommand*{\glxtrinlinefullformat}[2]{%
9962   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9963   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9964   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
9965 }%
9966 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9967   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9968   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9969   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
9970 }%
9971 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9972   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9973   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9974   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
9975 }%
9976 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9977   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9978   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9979   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
9980 }%

```

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.

```

9981 \renewcommand*{\glxtrfullformat}[2]{%
9982   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9983   \ifglxtrinsertinside\else##2\fi
9984 }%
9985 \renewcommand*{\glxtrfullplformat}[2]{%
9986   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9987   \ifglxtrinsertinside\else##2\fi
9988 }%
9989 \renewcommand*{\Glsxtrfullformat}[2]{%
9990   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9991   \ifglxtrinsertinside\else##2\fi
9992 }%

```

```

9993 \renewcommand*{\Glsxtrfullplformat}[2]{%
9994   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9995   \ifglxtrinsertinside\else##2\fi
9996 }%
9997 }

```

long-desc-sm Backward compatibility:

```

9998 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```

9999 \newabbreviationstyle{short-sm-footnote}%
10000 {%

```

Set accessibility attributes if enabled.

```

10001 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

10002 \renewcommand*{\CustomAbbreviationFields}{%
10003   name={\glxtrfootnotename},
10004   sort={\the\glsshorttok},
10005   description={\the\glslongtok},%
10006   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
10007     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10008     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
10009   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
10010     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10011     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
10012   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
10013   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.

```

10014 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10015   \glsetattribute{\the\glslabeltok}{nohyperfirst}{true}%
10016   \glshasattribute{\the\glslabeltok}{regular}%
10017   {%
10018     \glsetattribute{\the\glslabeltok}{regular}{false}%
10019   }%
10020   {}%
10021 }%
10022 }%
10023 {%

```

```

10024 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
10025 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
10026 \renewcommand*\abbrvpluralsuffix{\glxtrsmsuffix}%
10027 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
10028 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

10029 \renewcommand*{\glxtrfullformat}[2]{%
10030   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10031   \ifglxtrininsertinside\else##2\fi
10032   \protect\glxtrabbrvfootnote{##1}%
10033   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10034 }%
10035 \renewcommand*{\glxtrfullplformat}[2]{%
10036   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10037   \ifglxtrininsertinside\else##2\fi
10038   \protect\glxtrabbrvfootnote{##1}%
10039   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10040 }%
10041 \renewcommand*{\Glsxtrfullformat}[2]{%
10042   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10043   \ifglxtrininsertinside\else##2\fi
10044   \protect\glxtrabbrvfootnote{##1}%
10045   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10046 }%
10047 \renewcommand*{\Glsxtrfullplformat}[2]{%
10048   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10049   \ifglxtrininsertinside\else##2\fi
10050   \protect\glxtrabbrvfootnote{##1}%
10051   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10052 }%

```

The first use full form and the inline full form use the short (long) style.

```

10053 \renewcommand*{\glxtrininlinefullformat}[2]{%
10054   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10055   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10056   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10057 }%
10058 \renewcommand*{\glxtrininlinefullplformat}[2]{%
10059   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10060   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10061   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10062 }%
10063 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
10064   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10065   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10066   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10067 }%
10068 \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
10069   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10070   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10071   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10072 }%
10073 }

```

footnote-sm Backward compatibility:

```

10074 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

m-footnote-desc Like short-footnote but with user supplied description.

```
10075 \newabbreviationstyle{short-sm-footnote-desc}%
10076 {%
```

Set accessibility attributes if enabled.

```
10077 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```
10078 \renewcommand*{\CustomAbbreviationFields}{%
10079   name={\glxtrfootnotedesname},
10080   sort={\glxtrfootnotedesort},
10081   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
10082     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10083     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
10084   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
10085     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10086     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
10087   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
10088   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.

```
10089 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10090   \glsssetAttribute{\the\glslabeltok}{nohyperfirst}{true}%
10091   \glshasattribute{\the\glslabeltok}{regular}%
10092   {%
10093     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
10094   }%
10095   {}%
10096 }%
10097}%
10098{%
10099 \GlsXtrUseAbbrStyleFmts{short-sm-footnote}%
10100 }
```

sm-postfootnote

```
10101 \newabbreviationstyle{short-sm-postfootnote}%
10102 {%
```

Set accessibility attributes if enabled.

```
10103 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
10104 \renewcommand*{\CustomAbbreviationFields}{%
10105   name={\glxtrfootnotename},
10106   sort={\the\glsshorttok},
10107   description={\the\glslongtok},%
10108   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
10109   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
10110   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
10111   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.

```
10112 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10113 \csdef{glxstrpostlink\glscategorylabel}{%
10114 \glxtrifwasfirstuse
10115 }
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
10116 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
10117 {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
10118 }%
10119 {}%
10120 }%
10121 \glshasattribute{\the\glslabeltok}{regular}%
10122 {%
10123 \glissetattribute{\the\glslabeltok}{regular}{false}%
10124 }%
10125 {}%
10126 }%
```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
10127 \renewcommand*{\glxtrsetupfulldefs}{%
10128 \let\glxtrifwasfirstuse\@secondoftwo
10129 }%
10130 }%
10131 {%
```

```
10132 \renewcommand*\glssabbrvfont[1]{\glssabbrvsmfont{##1}}%
10133 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
10134 \renewcommand*\abbrvpluralsuffix{\glxtrsmsuffix}%
10135 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
10136 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
10137 \renewcommand*\glxtrfullformat[2]{%
10138 \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10139 \ifglxtrininsertinside\else##2\fi
10140 }%
10141 \renewcommand*\glxtrfullplformat[2]{%
10142 \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10143 \ifglxtrininsertinside\else##2\fi
10144 }%
10145 \renewcommand*\Glsxtrfullformat[2]{%
10146 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10147 \ifglxtrininsertinside\else##2\fi
10148 }%
10149 \renewcommand*\Glsxtrfullplformat[2]{%
10150 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

10151 \ifglxtrinsertinside\else##2\fi
10152 }%

```

The first use full form and the inline full form use the short (long) style.

```

10153 \renewcommand*\glxtrinlinefullformat}[2]{%
10154   \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10155   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10156   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
10157 }%
10158 \renewcommand*\glxtrinlinefullplformat}[2]{%
10159   \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10160   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10161   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
10162 }%
10163 \renewcommand*\Glsxtrinlinefullformat}[2]{%
10164   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10165   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10166   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
10167 }%
10168 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
10169   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10170   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10171   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
10172 }%
10173 }

```

postfootnote-sm Backward compatibility:

```

10174 \@glxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

stfootnote-desc Like short-sm-postfootnote but with user supplied description.

```

10175 \newabbreviationstyle{short-sm-postfootnote-desc}%
10176 {}

```

Set accessibility attributes if enabled.

```

10177 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

10178 \renewcommand*\CustomAbbreviationFields{%
10179   name={\glxtrfootnotedesname},
10180   sort={\glxtrfootnotedesort},
10181   first={\protect\glsfirstabbrvsmfont{\the\glssshorttok}},%
10182   firstplural={\protect\glsfirstabbrvsmfont{\the\glssshortpltok}},%
10183   text={\protect\glsabbrvsmfont{\the\glssshorttok}},%
10184   plural={\protect\glsabbrvsmfont{\the\glssshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

10185 \renewcommand*\GlsXtrPostNewAbbreviation{%
10186   \csdef{glxtrpostlink\glscategorylabel}{%
10187     \glxtrifwasfirstuse
10188     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

10189      \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
10190      {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%
10191      }%
10192      {}%
10193      }%
10194      \glshasattribute{\the\glslabeltok}{regular}%
10195      {%
10196      \glissetattribute{\the\glslabeltok}{regular}{false}%
10197      }%
10198      {}%
10199      }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

10200 \renewcommand*{\glxtrsetupfulldefs}{%
10201 \let\glxtrifwasfirstuse\@secondoftwo
10202 }%
10203 }%
10204 {%
10205 \GlsXtrUseAbbrStyleFmts{short-sm-postfootnote}%
10206 }

```

1.7.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```

10207 \newcommand*{\glsabbrvemfont}[1]{\emph{#1}}%

```

`\glfirstabbrvemfont`

```

10208 \newcommand*{\glfirstabbrvemfont}[1]{\glsabbrvemfont{#1}}%

```

The default short form suffix:

`\glxxtremsuffix`

```

10209 \newcommand*{\glxxtremsuffix}{\glxtrabbrvpluralsuffix}

```

`\glfirstlongemfont` Only used by the “long-em” styles.

```

10210 \newcommand*{\glfirstlongemfont}[1]{\glslongemfont{#1}}%

```

`\glslongemfont` Only used by the “long-em” styles.

```

10211 \newcommand*{\glslongemfont}[1]{\emph{#1}}%

```

`\long-short-em` The long form is just set in the default long font.

```

10212 \newabbreviationstyle{long-short-em}%
10213 {}

```

Set accessibility attributes if enabled.

```
10214 \glxstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
10215 \renewcommand*{\CustomAbbreviationFields}{%
10216   name={\glxstrlongshortname},
10217   sort={\the\glsshorttok},
10218   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
10219     \protect\glxstrfullsep{\the\glslabeltok}%
10220     \glxstrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10221   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
10222     \protect\glxstrfullsep{\the\glslabeltok}%
10223     \glxstrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
10224   text={\protect\glssabbrvemfont{\the\glsshorttok}},%
10225   plural={\protect\glssabbrvemfont{\the\glsshortpltok}},%
10226   description={\the\glslongtok}}%
10227 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10228   \glshalattribute{\the\glslabeltok}{regular}%
10229   {%
10230     \glissetattribute{\the\glslabeltok}{regular}{false}%
10231   }%
10232   {}%
10233 }%
10234 }%
10235 {%
```

```
10236 \renewcommand*\glssabbrvfont[1]{\glssabbrvemfont{##1}}%
10237 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
10238 \renewcommand*\abbrvpluralsuffix{\glsxtrlemsuffix}%
```

Use the default long fonts.

```
10239 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
10240 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
10241 \renewcommand*\glxstrfullformat[2]{%
10242   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxstrinsertinside##2\fi}%
10243   \ifglxstrinsertinside\else##2\fi
10244   \glxstrfullsep{##1}%
10245   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
10246 }%
10247 \renewcommand*\glxstrfullplformat[2]{%
10248   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxstrinsertinside##2\fi}%
10249   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10250   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
10251 }%
10252 \renewcommand*\Glsxtrfullformat[2]{%
10253   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxstrinsertinside##2\fi}%
10254   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10255   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%

```

```

10256 }%
10257 \renewcommand*{\Glsxtrfullplformat}[2]{%
10258   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10259   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10260   \glxtrparen{\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
10261 }%
10262 }

```

g-short-em-desc

```

10263 \newabbreviationstyle{long-short-em-desc}%
10264 {%

```

Set accessibility attributes if enabled.

```

10265   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

10266   \renewcommand*{\CustomAbbreviationFields}{%
10267     name={\glxtrlongshortdescname},
10268     sort={\glxtrlongshortdescsort},%
10269     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
10270       \protect\glxtrfullsep{\the\glslabeltok}%
10271       \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10272     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
10273       \protect\glxtrfullsep{\the\glslabeltok}%
10274       \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
10275     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10276     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
10277   }%

```

Unset the regular attribute if it has been set.

```

10278   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10279     \glshasattribute{\the\glslabeltok}{regular}%
10280     {%
10281       \glissetattribute{\the\glslabeltok}{regular}{false}%
10282     }%
10283     {}%
10284   }%
10285 }%
10286 {%

```

As long-short-em style:

```

10287   \GlsXtrUseAbbrStyleFmts{long-short-em}%
10288 }

```

ong-em-short-em

```

10289 \newabbreviationstyle{long-em-short-em}%
10290 {%

```

Set accessibility attributes if enabled.

```

10291   \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields. `\glslongemfont` is used in the description since `\glsdesc` doesn't set the style.

```

10292 \renewcommand*{\CustomAbbreviationFields}{%
10293     name={\glsxtrlongshortname},
10294     sort={\the\glsshorttok},
10295     first={\protect\glsfirstlongemfont{\the\glslongtok}%
10296         \protect\glsxtrfullsep{\the\glslabeltok}%
10297         \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10298     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
10299         \protect\glsxtrfullsep{\the\glslabeltok}%
10300         \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%

10301     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10302     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
10303     description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10304 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10305     \glshasattribute{\the\glslabeltok}{regular}%
10306     {%
10307         \glissetattribute{\the\glslabeltok}{regular}{false}%
10308     }%
10309     {}%
10310 }%
10311 }%
10312 {%

10313 \renewcommand*{\abbrvpluralsuffix}{\glsxtremsuffix}%
10314 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10315 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10316 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
10317 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10318 \renewcommand*{\glsxtrfullformat}[2]{%
10319     \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10320     \ifglsxtrininsertinside\else##2\fi
10321     \glsxtrfullsep{##1}%
10322     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10323 }%
10324 \renewcommand*{\glsxtrfullplformat}[2]{%
10325     \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10326     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10327     \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10328 }%
10329 \renewcommand*{\Glsxtrfullformat}[2]{%
10330     \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10331     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10332     \glsxtrparen{\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
10333 }%

```

```

10334 \renewcommand*{\Glsxtrfullplformat}[2]{%
10335   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10336   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10337   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10338 }%
10339 }

```

m-short-em-desc

```

10340 \newabbreviationstyle{long-em-short-em-desc}%
10341 {%
  Set accessibility attributes if enabled.
10342   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
  Setup the default fields.
10343   \renewcommand*{\CustomAbbreviationFields}{%
10344     name={\glsxtrlongshortdescname},
10345     sort={\glsxtrlongshortdescsort},%
10346     first={\protect\glsfirstlongemfont{\the\glslongtok}%
10347       \protect\glxtrfullsep{\the\glslabeltok}%
10348       \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10349     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
10350       \protect\glxtrfullsep{\the\glslabeltok}%
10351       \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
10352     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10353     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
10354   }%
  Unset the regular attribute if it has been set.
10355   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10356     \glshasattribute{\the\glslabeltok}{regular}%
10357     {%
10358       \glissetattribute{\the\glslabeltok}{regular}{false}%
10359     }%
10360     {}%
10361   }%
10362 }%
10363 {%
10364   \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
10365 }

```

short-em-long Now the short (long) version

```

10366 \newabbreviationstyle{short-em-long}%
10367 {%
  Set accessibility attributes if enabled.
10368   \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
  Setup the default fields.
10369   \renewcommand*{\CustomAbbreviationFields}{%
10370     name={\glsxtrshortlongname},

```

```

10371 sort={\the\glsshorttok},
10372 description={\the\glslongtok},%
10373 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
10374 \protect\glsxtrfullsep{\the\glslabeltok}%
10375 \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
10376 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10377 \protect\glsxtrfullsep{\the\glslabeltok}%
10378 \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
10379 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10380 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

10381 \renewcommand*\GlsXtrPostNewAbbreviation{%
10382 \glsattribute{\the\glslabeltok}{regular}%
10383 {%
10384 \glssetattribute{\the\glslabeltok}{regular}{false}%
10385 }%
10386 {}%
10387 }%
10388 }%
10389 {%

```

Mostly as short-long style:

```

10390 \renewcommand*\abbrvpluralsuffix{\glsxtrsuffix}%
10391 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
10392 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
10393 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
10394 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10395 \renewcommand*\glsxtrfullformat[2]{%
10396 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
10397 \ifglsxtrininsertinside\else##2\fi
10398 \glsxtrfullsep{##1}%
10399 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
10400 }%
10401 \renewcommand*\glsxtrfullplformat[2]{%
10402 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
10403 \ifglsxtrininsertinside\else##2\fi
10404 \glsxtrfullsep{##1}%
10405 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
10406 }%
10407 \renewcommand*\Glsxtrfullformat[2]{%
10408 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
10409 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10410 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
10411 }%
10412 \renewcommand*\Glsxtrfullplformat[2]{%
10413 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
10414 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10415 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%

```



```

10416 }%
10417 }

```

rt-em-long-desc As before but user provides description

```

10418 \newabbreviationstyle{short-em-long-desc}%
10419 {%

```

Set accessibility attributes if enabled.

```

10420 \glstrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

10421 \renewcommand*{\CustomAbbreviationFields}{%
10422   name={\glstrshortlongdescname},
10423   sort={\glstrshortlongdescsort},
10424   first={\protect\glstrfirstabrvfont{\the\glsshorttok}%
10425     \protect\glstrfullsep{\the\glslabeltok}%
10426     \glstrparen{\protect\glstrfirstlongdefaultfont{\the\glslongtok}}},%
10427   firstplural={\protect\glstrfirstabrvfont{\the\glsshortpltok}%
10428     \protect\glstrfullsep{\the\glslabeltok}%
10429     \glstrparen{\protect\glstrfirstlongdefaultfont{\the\glslongpltok}}},%
10430   text={\protect\glstrabrvfont{\the\glsshorttok}},%
10431   plural={\protect\glstrabrvfont{\the\glsshortpltok}}}%
10432 }%

```

Unset the regular attribute if it has been set.

```

10433 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10434   \glshasattribute{\the\glslabeltok}{regular}%
10435   {%
10436     \glissetattribute{\the\glslabeltok}{regular}{false}%
10437   }%
10438   {}%
10439 }%
10440 }%
10441 {%
10442   \GlsXtrUseAbbrStyleFmts{short-em-long}%
10443 }

```

hort-em-long-em

```

10444 \newabbreviationstyle{short-em-long-em}%
10445 {%

```

Set accessibility attributes if enabled.

```

10446 \glstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields. \glslongemfont is used in the description since \glsdsc doesn't set the style.

```

10447 \renewcommand*{\CustomAbbreviationFields}{%
10448   name={\glstrshortlongname},
10449   sort={\the\glsshorttok},
10450   description={\protect\glslongemfont{\the\glslongtok}},%
10451   first={\protect\glstrfirstabrvfont{\the\glsshorttok}%

```

```

10452 \protect\glxtrfullsep{\the\glslabeltok}%
10453 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
10454 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10455 \protect\glxtrfullsep{\the\glslabeltok}%
10456 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%

10457 text={\protect\glsabbrvemfont{\the\glsshorttok}}},%
10458 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}},%

```

Unset the regular attribute if it has been set.

```

10459 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10460 \glshasattribute{\the\glslabeltok}{regular}}%
10461 {%
10462 \glissetattribute{\the\glslabeltok}{regular}{false}}%
10463 }%
10464 {}%
10465 }%
10466 }%
10467 {%

10468 \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10469 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10470 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10471 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
10472 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10473 \renewcommand*{\glxtrfullformat}[2]{%
10474 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10475 \ifglxtrininsertinside\else##2\fi
10476 \glxtrfullsep{##1}%
10477 \glxtrparen{\glsfirstlongemfont{\glssaccesslong{##1}}}%
10478 }%
10479 \renewcommand*{\glxtrfullplformat}[2]{%
10480 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10481 \ifglxtrininsertinside\else##2\fi
10482 \glxtrfullsep{##1}%
10483 \glxtrparen{\glsfirstlongemfont{\glssaccesslongpl{##1}}}%
10484 }%
10485 \renewcommand*{\Glsxtrfullformat}[2]{%
10486 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10487 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10488 \glxtrparen{\glsfirstlongemfont{\glssaccesslong{##1}}}%
10489 }%
10490 \renewcommand*{\Glsxtrfullplformat}[2]{%
10491 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10492 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10493 \glxtrparen{\glsfirstlongemfont{\glssaccesslongpl{##1}}}%
10494 }%
10495 }

```

em-long-em-desc

```
10496 \newabbreviationstyle{short-em-long-em-desc}%
10497 {%
```

Set accessibility attributes if enabled.

```
10498 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```
10499 \renewcommand*{\CustomAbbreviationFields}{%
10500   name={\glxtrshortlongdescname},%
10501   sort={\glxtrshortlongdescsort},%
10502   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
10503     \protect\glxtrfullsep{\the\glslabeltok}%
10504     \glxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
10505   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10506     \protect\glxtrfullsep{\the\glslabeltok}%
10507     \glxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
10508   text={\protect\glssabbrvemfont{\the\glsshorttok}},%
10509   plural={\protect\glssabbrvemfont{\the\glsshortpltok}}}%
10510 }
```

Unset the regular attribute if it has been set.

```
10511 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10512   \glshasattribute{\the\glslabeltok}{regular}%
10513   {%
10514     \glissetattribute{\the\glslabeltok}{regular}{false}%
10515   }%
10516   {}%
10517 }%
10518 }%
10519 {%
10520   \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
10521 }
```

short-em

```
10522 \newabbreviationstyle{short-em}%
10523 {%
```

Set accessibility attributes if enabled.

```
10524 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
10525 \renewcommand*{\CustomAbbreviationFields}{%
10526   name={\glxtrshortnolongname},
10527   sort={\the\glsshorttok},
10528   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
10529   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
10530   text={\protect\glssabbrvemfont{\the\glsshorttok}},
10531   plural={\protect\glssabbrvemfont{\the\glsshortpltok}},
10532   description={\the\glslongtok}}%
10533 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```

10534 \glsssetAttribute{\the\glslabelltok}{regular}{true}}%
10535 }%
10536 {%

```

```

10537 \renewcommand*{\abbrvpluralsuffix}{\glstremssuffix}%
10538 \renewcommand*{\glssabrvfont}[1]{\glssabrvfont{##1}}%
10539 \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvfont{##1}}%
10540 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
10541 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

10542 \renewcommand*{\glssxtrinlinefullformat}[2]{%
10543 \protect\glssfirstabrvfont{\glssaccesssshort{##1}}%
10544 \ifglssxtrininsertinside##2\fi}%
10545 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}}%
10546 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslong{##1}}}%
10547 }%
10548 \renewcommand*{\glssxtrinlinefullplformat}[2]{%
10549 \protect\glssfirstabrvfont{\glssaccesssshortpl{##1}}%
10550 \ifglssxtrininsertinside##2\fi}%
10551 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}}%
10552 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
10553 }%

10554 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10555 \protect\glssfirstabrvfont{\Glsaccesssshort{##1}}%
10556 \ifglssxtrininsertinside##2\fi}%
10557 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}}%
10558 \glssxtrparen{\glssfirstlongdefaultfont{\Glsaccesslong{##1}}}%
10559 }%
10560 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10561 \protect\glssfirstabrvfont{\Glsaccesssshortpl{##1}}%
10562 \ifglssxtrininsertinside##2\fi}%
10563 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}}%
10564 \glssxtrparen{\glssfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
10565 }%

```

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.

```

10566 \renewcommand*{\glssxtrfullformat}[2]{%
10567 \glssfirstabrvfont{\glssaccesssshort{##1}}\ifglssxtrininsertinside##2\fi}%
10568 \ifglssxtrininsertinside\else##2\fi
10569 }%
10570 \renewcommand*{\glssxtrfullplformat}[2]{%
10571 \glssfirstabrvfont{\glssaccesssshortpl{##1}}\ifglssxtrininsertinside##2\fi}%
10572 \ifglssxtrininsertinside\else##2\fi
10573 }%
10574 \renewcommand*{\Glsxtrfullformat}[2]{%
10575 \glssfirstabrvfont{\glssaccesssshort{##1}}\ifglssxtrininsertinside##2\fi}%
10576 \ifglssxtrininsertinside\else##2\fi

```

```

10577 }%
10578 \renewcommand*{\Glsxtrfullplformat}[2]{%
10579   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10580   \ifglxtrinsertinside\else##2\fi
10581 }%
10582 }

```

short-em-nolong

```

10583 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

10584 \newabbreviationstyle{short-em-desc}%
10585 {%

```

Set accessibility attributes if enabled. The default name includes the long form but `\glsxtrshortdescname` could be modified to omit the long form, so include the `nameshortaccess` attribute.

```

10586   \glsxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

10587 \renewcommand*{\CustomAbbreviationFields}{%
10588   name={\glsxtrshortdescname},
10589   sort={\the\glsshorttok},
10590   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
10591   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
10592   text={\protect\glsabbrvemfont{\the\glsshorttok}},
10593   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
10594 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10595   \glssetattribute{\the\glslabeltok}{regular}{true}}%
10596 }%
10597 {%

```

```

10598   \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10599   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10600   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10601   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
10602   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

10603 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10604   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10605   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
10606   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
10607 }%
10608 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10609   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10610   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
10611   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
10612 }%
10613 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10614   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%

```

```

10615 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10616 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
10617 }%
10618 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10619 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10620 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10621 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
10622 }%

```

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.

```

10623 \renewcommand*{\glxtrfullformat}[2]{%
10624 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10625 \ifglxtrinsertinside\else##2\fi
10626 }%
10627 \renewcommand*{\glxtrfullplformat}[2]{%
10628 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10629 \ifglxtrinsertinside\else##2\fi
10630 }%
10631 \renewcommand*{\Glsxtrfullformat}[2]{%
10632 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10633 \ifglxtrinsertinside\else##2\fi
10634 }%
10635 \renewcommand*{\Glsxtrfullplformat}[2]{%
10636 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10637 \ifglxtrinsertinside\else##2\fi
10638 }%
10639 }

```

-em-nolong-desc

```

10640 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

10641 \newabbreviationstyle{nolong-short-em}%
10642 {%
10643 \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
10644 }%
10645 {%
10646 \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

10647 \renewcommand*{\glxtrinlinefullformat}[2]{%
10648 \protect\glsfirstlongdefaultfont{\glssaccesslong{##1}%
10649 \ifglxtrinsertinside##2\fi}%
10650 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10651 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
10652 }%
10653 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10654 \protect\glsfirstlongdefaultfont{\glssaccesslongpl{##1}%
10655 \ifglxtrinsertinside##2\fi}%

```

```

10656 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10657 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10658 }%
10659 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10660 \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
10661 \ifglxtrinsertinside##2\fi}%
10662 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10663 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10664 }%
10665 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10666 \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
10667 \ifglxtrinsertinside##2\fi}%
10668 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10669 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10670 }%
10671 }

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

10672 \newabbreviationstyle{long-noshort-em}%
10673 {%

```

Set accessibility attributes if enabled.

```
10674 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel
```

Setup the default fields.

```

10675 \renewcommand*{\CustomAbbreviationFields}{%
10676 name={\glxtrlongnoshortname},
10677 sort={\the\glsshorttok},
10678 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
10679 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
10680 text={\protect\glslongdefaultfont{\the\glslongtok}},
10681 plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
10682 description={\the\glslongtok}%
10683 }%
10684 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10685 \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
10686 }%
10687 {%

```

```

10688 \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10689 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10690 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10691 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
10692 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10693 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10694 \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
10695 \ifglxtrinsertinside \else##2\fi
10696 }%
10697 \renewcommand*{\glxtrsubsequentplfmt}[2]{%

```

```

10698 \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
10699 \ifglxtrinsertinside \else##2\fi
10700 }%
10701 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10702 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
10703 \ifglxtrinsertinside \else##2\fi
10704 }%
10705 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10706 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
10707 \ifglxtrinsertinside \else##2\fi
10708 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10709 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10710 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10711 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
10712 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10713 }%
10714 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10715 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10716 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
10717 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10718 }%
10719 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10720 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10721 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
10722 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10723 }%
10724 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10725 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10726 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
10727 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10728 }%

```

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.

```

10729 \renewcommand*{\glsxtrfullformat}[2]{%
10730 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10731 \ifglxtrinsertinside\else##2\fi
10732 }%
10733 \renewcommand*{\glsxtrfullplformat}[2]{%
10734 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10735 \ifglxtrinsertinside\else##2\fi
10736 }%
10737 \renewcommand*{\Glsxtrfullformat}[2]{%
10738 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10739 \ifglxtrinsertinside\else##2\fi
10740 }%
10741 \renewcommand*{\Glsxtrfullplformat}[2]{%
10742 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%

```



```

10743 \ifglxtrinsertinside\else##2\fi
10744 }%
10745 }

```

long-em Backward compatibility:

```

10746 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}

```

g-em-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

10747 \newabbreviationstyle{long-em-noshort-em}%
10748 {%

```

Set accessibility attributes if enabled.

```

10749 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```

10750 \renewcommand*{\CustomAbbreviationFields}{%
10751   name={\glxtrlongnoshortname},
10752   sort={\the\glsshorttok},
10753   first={\protect\glfirstlongemfont{\the\glslongtok}},
10754   firstplural={\protect\glfirstlongemfont{\the\glslongpltok}},
10755   text={\protect\glslongemfont{\the\glslongtok}},
10756   plural={\protect\glslongemfont{\the\glslongpltok}},%
10757   description={\protect\glslongemfont{\the\glslongtok}}%
10758 }%
10759 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10760   \glissetattribute{\the\glslabeltok}{regular}{true}}%
10761 }%
10762 {%

```

```

10763 \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10764 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10765 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvemfont{##1}}%
10766 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongemfont{##1}}%
10767 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10768 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10769   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
10770   \ifglxtrinsertinside \else##2\fi
10771 }%
10772 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10773   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
10774   \ifglxtrinsertinside \else##2\fi
10775 }%
10776 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10777   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
10778   \ifglxtrinsertinside \else##2\fi
10779 }%
10780 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10781   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%

```

```

10782 \ifglxtrinsertinside \else##2\fi
10783 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10784 \renewcommand*{\glxtrinlinefullformat}[2]{%
10785   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10786   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10787   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10788 }%
10789 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10790   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10791   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10792   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10793 }%
10794 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10795   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10796   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10797   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10798 }%
10799 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10800   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10801   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10802   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10803 }%

```

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.

```

10804 \renewcommand*{\glxtrfullformat}[2]{%
10805   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10806   \ifglxtrinsertinside\else##2\fi
10807 }%
10808 \renewcommand*{\glxtrfullplformat}[2]{%
10809   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10810   \ifglxtrinsertinside\else##2\fi
10811 }%
10812 \renewcommand*{\Glsxtrfullformat}[2]{%
10813   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10814   \ifglxtrinsertinside\else##2\fi
10815 }%
10816 \renewcommand*{\Glsxtrfullplformat}[2]{%
10817   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10818   \ifglxtrinsertinside\else##2\fi
10819 }%
10820 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

10821 \newabbreviationstyle{long-em-noshort-em-noreg}%
10822 {}%

```

Set accessibility attributes if enabled.

```

10823 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```
10824 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%
```

Unset the regular attribute if it has been set.

```
10825 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
10826 \glshasattribute{\the\glslabeltok}{regular}}%  
10827 {%  
10828 \glissetattribute{\the\glslabeltok}{regular}{false}}%  
10829 }%  
10830 {}%  
10831 }%  
10832 }%  
10833 {%  
10834 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%  
10835 }
```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
10836 \newabbreviationstyle{long-noshort-em-desc}%  
10837 {%  
10838 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%  
10839 }%  
10840 {%
```

```
10841 \renewcommand*{\abbrvpluralsuffix}{\glstremsuffix}%  
10842 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%  
10843 \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvemfont{##1}}%  
10844 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%  
10845 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
10846 \renewcommand*{\glsxtrsubsequentfmt}[2]{%  
10847 \glsslongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%  
10848 \ifglsxtrininsertinside \else##2\fi  
10849 }%  
10850 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%  
10851 \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%  
10852 \ifglsxtrininsertinside \else##2\fi  
10853 }%  
10854 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%  
10855 \glsslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%  
10856 \ifglsxtrininsertinside \else##2\fi  
10857 }%  
10858 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%  
10859 \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%  
10860 \ifglsxtrininsertinside \else##2\fi  
10861 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
10862 \renewcommand*{\glsxtrininlinefullformat}[2]{%
```

```

10863 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10864 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10865 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10866 }%
10867 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10868 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10869 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10870 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10871 }%
10872 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10873 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10874 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10875 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10876 }%
10877 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10878 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10879 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10880 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10881 }%

```

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.

```

10882 \renewcommand*{\glxtrfullformat}[2]{%
10883 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10884 \ifglxtrinsertinside\else##2\fi
10885 }%
10886 \renewcommand*{\glxtrfullplformat}[2]{%
10887 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10888 \ifglxtrinsertinside\else##2\fi
10889 }%
10890 \renewcommand*{\Glsxtrfullformat}[2]{%
10891 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10892 \ifglxtrinsertinside\else##2\fi
10893 }%
10894 \renewcommand*{\Glsxtrfullplformat}[2]{%
10895 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10896 \ifglxtrinsertinside\else##2\fi
10897 }%
10898 }

```

long-desc-em Backward compatibility:

```

10899 \@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.

```

10900 \newabbreviationstyle{long-em-noshort-em-desc}%
10901 {%
10902 \renewcommand*{\CustomAbbreviationFields}{%
10903 name={\glxtrlongnoshortdescname},

```

```

10904     sort={\the\glslongtok},
10905     first={\protect\glsfirstlongemfont{\the\glslongtok}},
10906     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
10907     text={\glslongemfont{\the\glslongtok}},
10908     plural={\glslongemfont{\the\glslongpltok}}%
10909 }%
10910 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10911   \glssetattribute{\the\glslabeltok}{regular}{true}}%
10912 }%
10913 {%
10914   \renewcommand*{\abbrvpluralsuffix}{\glsxtremsuffix}%
10915   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10916   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10917   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
10918   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10919   \renewcommand*{\glsxtrsubsequentfmt}[2]{%
10920     \glslongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
10921     \ifglsxtrininsertinside \else##2\fi
10922   }%
10923   \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
10924     \glslongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
10925     \ifglsxtrininsertinside \else##2\fi
10926   }%
10927   \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10928     \glslongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
10929     \ifglsxtrininsertinside \else##2\fi
10930   }%
10931   \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10932     \glslongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
10933     \ifglsxtrininsertinside \else##2\fi
10934   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10935   \renewcommand*{\glsxtrinlinefullformat}[2]{%
10936     \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10937     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10938     \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10939   }%
10940   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10941     \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10942     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10943     \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10944   }%
10945   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10946     \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10947     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10948     \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%

```

```

10949 }%
10950 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10951   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10952   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10953   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
10954 }%

```

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.

```

10955 \renewcommand*{\glxtrfullformat}[2]{%
10956   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10957   \ifglxtrinsertinside\else##2\fi
10958 }%
10959 \renewcommand*{\glxtrfullplformat}[2]{%
10960   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10961   \ifglxtrinsertinside\else##2\fi
10962 }%
10963 \renewcommand*{\Glsxtrfullformat}[2]{%
10964   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10965   \ifglxtrinsertinside\else##2\fi
10966 }%
10967 \renewcommand*{\Glsxtrfullplformat}[2]{%
10968   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10969   \ifglxtrinsertinside\else##2\fi
10970 }%
10971 }

```

`t-em-desc-noreg` Like `long-em-noshort-em-desc` but doesn't set the regular attribute.

```

10972 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
10973 {%
10974   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%

```

Unset the regular attribute if it has been set.

```

10975 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10976   \glshasattribute{\the\glslabeltok}{regular}%
10977   {%
10978     \glissetattribute{\the\glslabeltok}{regular}{false}%
10979   }%
10980   {}}%
10981 }%
10982 }%
10983 {%
10984   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
10985 }

```

`ort-em-footnote`

```

10986 \newabbreviationstyle{short-em-footnote}%
10987 {%

```

Set accessibility attributes if enabled.

```

10988   \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

10989 \renewcommand*{\CustomAbbreviationFields}{%
10990     name={\glxtrfootnotename},
10991     sort={\the\glsshorttok},
10992     description={\the\glslongtok},%
10993     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
10994         \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10995         {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
10996     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10997         \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10998         {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
10999     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
11000     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.

```

11001 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11002     \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
11003     \glsattribute{\the\glslabeltok}{regular}%
11004     {%
11005         \glssetattribute{\the\glslabeltok}{regular}{false}%
11006     }%
11007     {}%
11008 }%
11009 }%
11010 {%

11011 \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
11012 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
11013 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
11014 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
11015 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

11016 \renewcommand*{\glxtrfullformat}[2]{%
11017     \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
11018     \ifglxtrininsertinside\else##2\fi
11019     \protect\glxtrabbrvfootnote{##1}%
11020     {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11021 }%
11022 \renewcommand*{\glxtrfullplformat}[2]{%
11023     \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
11024     \ifglxtrininsertinside\else##2\fi
11025     \protect\glxtrabbrvfootnote{##1}%
11026     {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11027 }%
11028 \renewcommand*{\Glsxtrfullformat}[2]{%
11029     \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
11030     \ifglxtrininsertinside\else##2\fi
11031     \protect\glxtrabbrvfootnote{##1}%

```

```

11032     {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11033 }%
11034 \renewcommand*{\Glsxtrfullplformat}[2]{%
11035     \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11036     \ifglxtrinsertinside\else##2\fi
11037     \protect\glxtrabbrvfootnote{##1}%
11038     {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11039 }%

```

The first use full form and the inline full form use the short (long) style.

```

11040 \renewcommand*{\glsxtrinlinefullformat}[2]{%
11041     \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11042     \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
11043     \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11044 }%
11045 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
11046     \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11047     \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
11048     \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11049 }%
11050 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11051     \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11052     \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
11053     \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11054 }%
11055 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11056     \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11057     \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
11058     \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11059 }%
11060 }

```

footnote-em Backward compatibility:

```

11061 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

m-footnote-desc Like short-em-footnote but with user supplied description.

```

11062 \newabbreviationstyle{short-em-footnote-desc}%
11063 {}

```

Set accessibility attributes if enabled.

```

11064 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

11065 \renewcommand*{\CustomAbbreviationFields}{%
11066     name={\glxtrfootnotedesname},
11067     sort={\glxtrfootnotedesort},
11068     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
11069         \protect\glxtrabbrvfootnote{\the\glslabeltok}%
11070         {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
11071     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%

```



```

11072 \protect\glstrabbrvfootnote{\the\glslabeltok}%
11073 {\protect\glslongfootnotefont{\the\glslongpltok}}},%
11074 text={\protect\glssabrvemfont{\the\glsshorttok}},%
11075 plural={\protect\glssabrvemfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

11076 \renewcommand*\GlsXtrPostNewAbbreviation{%
11077 \glsssetAttribute{\the\glslabeltok}{nohyperfirst}{true}%
11078 \glshasattribute{\the\glslabeltok}{regular}%
11079 {%
11080 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
11081 }%
11082 {}%
11083 }%
11084}%
11085{%
11086 \GlsXtrUseAbbrStyleFmts{short-em-footnote}%
11087}

```

em-postfootnote

```

11088 \newabbreviationstyle{short-em-postfootnote}%
11089 {%

```

Set accessibility attributes if enabled.

```

11090 \glstrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

11091 \renewcommand*\CustomAbbreviationFields{%
11092 name={\glstrfootnotename},
11093 sort={\the\glsshorttok},
11094 description={\the\glslongtok},%
11095 first={\protect\glslongfootnotefont{\the\glsshorttok}},%
11096 firstplural={\protect\glssabrvemfont{\the\glsshortpltok}},%
11097 text={\protect\glssabrvemfont{\the\glsshorttok}},%
11098 plural={\protect\glssabrvemfont{\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.

```

11099 \renewcommand*\GlsXtrPostNewAbbreviation{%
11100 \csdef{glstrpostlink\glscategorylabel}{%
11101 \glstrifwasfirstuse
11102 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

11103 \glstrdopostpunc{\protect\glstrabbrvfootnote{\glslabel}%
11104 {\glslongfootnotefont{\glslong{\glslabel}}}}}%
11105 }%
11106 {}%
11107 }%

```

```

11108 \glshasattribute{\the\glslabeltok}{regular}%
11109 {%
11110 \glissetattribute{\the\glslabeltok}{regular}{false}%
11111 }%
11112 {}%
11113 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

11114 \renewcommand*\glxtrsetupfulldefs{%
11115 \let\glxtrifwasfirstuse\@secondoftwo
11116 }%
11117}%
11118{%

11119 \renewcommand*\glxtrpluralsuffix{\glxtremsuffix}%
11120 \renewcommand*\glxabbrvfont[1]{\glxabbrvemfont{##1}}%
11121 \renewcommand*\glxfirstabbrvfont[1]{\glxfirstabbrvemfont{##1}}%
11122 \renewcommand*\glxfirstlongfont[1]{\glxfirstlongfootnotefont{##1}}%
11123 \renewcommand*\glxlongfont[1]{\glxlongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

11124 \renewcommand*\glxtrfullformat[2]{%
11125 \glxfirstabbrvemfont{\glxaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11126 \ifglxtrinsertinside\else##2\fi
11127}%
11128 \renewcommand*\glxtrfullplformat[2]{%
11129 \glxfirstabbrvemfont{\glxaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11130 \ifglxtrinsertinside\else##2\fi
11131}%
11132 \renewcommand*\Glsxtrfullformat[2]{%
11133 \glxfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11134 \ifglxtrinsertinside\else##2\fi
11135}%
11136 \renewcommand*\Glsxtrfullplformat[2]{%
11137 \glxfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11138 \ifglxtrinsertinside\else##2\fi
11139}%

```

The first use full form and the inline full form use the short (long) style.

```

11140 \renewcommand*\glxtrinlinefullformat[2]{%
11141 \glxfirstabbrvemfont{\glxaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11142 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
11143 \glxtrparen{\glxfirstlongfootnotefont{\glxaccesslong{##1}}}%
11144}%
11145 \renewcommand*\glxtrinlinefullplformat[2]{%
11146 \glxfirstabbrvemfont{\glxaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11147 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
11148 \glxtrparen{\glxfirstlongfootnotefont{\glxaccesslongpl{##1}}}%
11149}%
11150 \renewcommand*\Glsxtrinlinefullformat[2]{%

```

```

11151 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11152 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
11153 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
11154 }%
11155 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11156 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11157 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
11158 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
11159 }%
11160 }

```

postfootnote-em Backward compatibility:

```

11161 \@glxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}

```

stfootnote-desc Like short-em-postfootnote but with user supplied description.

```

11162 \newabbreviationstyle{short-em-postfootnote-desc}%
11163 {%

```

Set accessibility attributes if enabled.

```

11164 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

11165 \renewcommand*{\CustomAbbreviationFields}{%
11166 name={\glxtrfootnotedesname},
11167 sort={\glxtrfootnotedesort},
11168 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},%
11169 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},%
11170 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
11171 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.

```

11172 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11173 \csdef{glxtrpostlink\glscategorylabel}{%
11174 \glxtrifwasfirstuse
11175 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

11176 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
11177 {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
11178 }%
11179 {}%
11180 }%
11181 \glshasattribute{\the\glslabeltok}{regular}%
11182 {%
11183 \glissetattribute{\the\glslabeltok}{regular}{false}%
11184 }%
11185 {}%
11186 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

11187 \renewcommand*{\glxtrsetupfulldefs}{%
11188   \let\glxtrifwasfirstuse\@secondoftwo
11189 }%
11190}%
11191{%
11192 \GlsXtrUseAbbrStyleFmts{short-em-postfootnote}%
11193 }

```

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:

`\glxtruserfield` Default is the `useri` field.

```

11194 \newcommand*{\glxtruserfield}{useri}

```

`\glxtruserparen` The format of the parenthetical information. The first argument is the long/short form. The second argument is the entry's label. If `\glscurrentfieldvalue` has been defined, then we have at least glossaries v4.23, which makes it easier for the user to adjust this.

```

11195 \ifdef\glscurrentfieldvalue
11196 {
11197   \newcommand*{\glxtruserparen}[2]{%
11198     \glxtrfullsep{#2}%
11199     \glxtrparen
11200     {#1\ifglshasfield{\glxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
11201   }
11202 }
11203 {
11204   \newcommand*{\glxtruserparen}[2]{%
11205     \glxtrfullsep{#2}%
11206     \glxtrparen
11207     {#1\ifglshasfield{\glxtruserfield}{#2}{, \@glo@thisvalue}{}}%
11208   }
11209 }

```

Font used for short form:

`\lsabbrvuserfont`

```

11210 \newcommand*{\lsabbrvuserfont}[1]{\glsabbrvdefaultfont{#1}}

```

Font used for short form on first use:

`\stabbrvuserfont`

```

11211 \newcommand*{\stabbrvuserfont}[1]{\glsabbrvuserfont{#1}}

```

Font used for long form:

glslonguserfont

```
11212 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}
```

Font used for long form on first use:

rstlonguserfont

```
11213 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
11214 \newcommand*{\glsxtrusersuffix}{\glsxtrabbrvpluralsuffix}
```

Description encapsulator.

userdescription The first argument is the description. The second argument is the label.

```
11215 \newcommand*{\glsuserdescription}[2]{\glslonguserfont{#1}}
```

long-short-user

```
11216 \newabbreviationstyle{long-short-user}{%
```

```
11217 {%
```

Set accessibility attributes if enabled.

```
11218 \glsxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
11219 \renewcommand*{\CustomAbbreviationFields}{%
```

```
11220 name={\glsxtrlongshortname},
```

```
11221 sort={\the\glsshorttok},
```

```
11222 first={\protect\glsfirstlonguserfont{\the\glslongtok}}%
```

```
11223 \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
```

```
11224 {\the\glslabeltok}}%,
```

```
11225 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}}%
```

```
11226 \protect\glsxtruserparen
```

```
11227 {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}}%,
```

```
11228 text={\protect\glsabbrvuserfont{\the\glsshorttok}}%,
```

```
11229 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%,
```

```
11230 description={\protect\glsuserdescription{\the\glslongtok}}%
```

```
11231 {\the\glslabeltok}}}%
```

Unset the regular attribute if it has been set.

```
11232 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
11233 \glshasattribute{\the\glslabeltok}{regular}}%
```

```
11234 {%
```

```
11235 \glissetattribute{\the\glslabeltok}{regular}{false}}%
```

```
11236 }%
```

```
11237 {}%
```

```
11238 }%
```

```
11239 }%
```

```
11240 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```

11241 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
11242 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
11243 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
11244 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
11245 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

11246 \renewcommand*{\glxtrfullformat}[2]{%
11247   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11248   \ifglxtrininsertinside\else##2\fi
11249   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
11250 }%
11251 \renewcommand*{\glxtrfullplformat}[2]{%
11252   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11253   \ifglxtrininsertinside\else##2\fi
11254   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
11255 }%
11256 \renewcommand*{\Glsxtrfullformat}[2]{%
11257   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11258   \ifglxtrininsertinside\else##2\fi
11259   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
11260 }%
11261 \renewcommand*{\Glsxtrfullplformat}[2]{%
11262   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11263   \ifglxtrininsertinside\else##2\fi
11264   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
11265 }%
11266 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

11267 \newabbreviationstyle{long-postshort-user}%
11268 {}

```

Set accessibility attributes if enabled.

```

11269 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

11270 \renewcommand*{\CustomAbbreviationFields}{%
11271   name={\glxtrlongshortname},
11272   sort={\the\glsshorttok},
11273   first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
11274   firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
11275   text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11276   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
11277   description={\protect\glsuserdescription{\the\glslongtok}%
11278     {\the\glslabeltok}}}%
11279 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11280   \csdef{glxtrpostlink\glscategorylabel}{%

```

```

11281     \glxtrifwasfirstuse
11282     {%
11283     \glxtruserparen
11284     {\glsfirstabbrvuserfont{\gl Sentryshort{\glslabel}}}%
11285     {\glslabel}%
11286     }%
11287     {}%
11288     }%
11289     \glshasattribute{\the\glslabeltok}{regular}%
11290     {%
11291     \glissetattribute{\the\glslabeltok}{regular}{false}%
11292     }%
11293     {}%
11294     }%
11295 }%
11296 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11297 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
11298 \renewcommand*{\glabbrvfont}[1]{\glabbrvuserfont{##1}}%
11299 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
11300 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
11301 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

11302 \renewcommand*{\glxtrfullformat}[2]{%
11303     \glsfirstlonguserfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11304     \ifglxtrininsertinside\else##2\fi
11305 }%
11306 \renewcommand*{\glxtrfullplformat}[2]{%
11307     \glsfirstlonguserfont{\glaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11308     \ifglxtrininsertinside\else##2\fi
11309 }%
11310 \renewcommand*{\Glsxtrfullformat}[2]{%
11311     \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11312     \ifglxtrininsertinside\else##2\fi
11313 }%
11314 \renewcommand*{\Glsxtrfullplformat}[2]{%
11315     \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
11316     \ifglxtrininsertinside\else##2\fi
11317 }%

```

In-line format:

```

11318 \renewcommand*{\glxtrininlinefullformat}[2]{%
11319     \glsfirstlonguserfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
11320     \ifglxtrininsertinside\else##2\fi
11321     \glxtruserparen{\glsfirstabbrvuserfont{\glaccessshort{##1}}}{##1}%
11322 }%
11323 \renewcommand*{\glxtrininlinefullplformat}[2]{%
11324     \glsfirstlonguserfont{\glaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

11325 \ifglxtrinsertinside\else##2\fi
11326 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
11327 }%
11328 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11329 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11330 \ifglxtrinsertinside\else##2\fi
11331 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
11332 }%
11333 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11334 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11335 \ifglxtrinsertinside\else##2\fi
11336 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
11337 }%
11338 }

```

ortuserdescname

```

11339 \newcommand*{\glxtrlongshortuserdescname}{%
11340 \protect\glslonguserfont{\the\glslongtok}%
11341 \protect\glxtruserparen
11342 {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}%
11343 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

11344 \newabbreviationstyle{long-postshort-user-desc}%
11345 {%

```

Set accessibility attributes if enabled.

```

11346 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

11347 \renewcommand*{\CustomAbbreviationFields}{%
11348 name={\glxtrlongshortuserdescname},
11349 sort={\the\glslongtok},
11350 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
11351 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

11352 text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11353 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
11354 }%
11355 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11356 \csdef{glxtrpostlink\glscategorylabel}{%
11357 \glxtrifwasfirstuse
11358 {%
11359 \glxtruserparen
11360 {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
11361 {\glslabel}%
11362 }%
11363 }%
11364 }%
11365 \glshasattribute{\the\glslabeltok}{regular}%

```



```

11366   {%
11367     \glsetattribute{\the\glslabeltok}{regular}{false}%
11368   }%
11369   {}%
11370 }%
11371 }%
11372 {%
11373   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
11374 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

11375 \newabbreviationstyle{short-postlong-user}%
11376 {%

```

Set accessibility attributes if enabled.

```

11377   \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

11378   \renewcommand*{\CustomAbbreviationFields}{%
11379     name={\glxtrshortlongname},
11380     sort={\the\glsshorttok},
11381     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
11382     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

11383     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11384     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
11385     description={\protect\glsuserdescription{\the\glslongtok}%
11386       {\the\glslabeltok}}}%
11387   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11388     \csdef{glxtrpostlink\glscategorylabel}{%
11389       \glxtrifwasfirstuse
11390       {%
11391         \glxtruserparen
11392         {\glsfirstlonguserfont{\gl Sentrylong{\glslabel}}}%
11393         {\glslabel}%
11394       }%
11395     }%
11396   }%
11397   \glshasattribute{\the\glslabeltok}{regular}%
11398   {%
11399     \glsetattribute{\the\glslabeltok}{regular}{false}%
11400   }%
11401   {}%
11402 }%
11403 }%
11404 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11405   \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
11406   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
11407   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%

```

```

11408 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
11409 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

11410 \renewcommand*{\glsxtrfullformat}[2]{%
11411   \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11412   \ifglsxtrininsertinside\else##2\fi
11413 }%
11414 \renewcommand*{\glsxtrfullplformat}[2]{%
11415   \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11416   \ifglsxtrininsertinside\else##2\fi
11417 }%
11418 \renewcommand*{\Glsxtrfullformat}[2]{%
11419   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11420   \ifglsxtrininsertinside\else##2\fi
11421 }%
11422 \renewcommand*{\Glsxtrfullplformat}[2]{%
11423   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11424   \ifglsxtrininsertinside\else##2\fi
11425 }%

```

In-line format:

```

11426 \renewcommand*{\glsxtrinlinefullformat}[2]{%
11427   \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11428   \ifglsxtrininsertinside\else##2\fi
11429   \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
11430 }%
11431 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
11432   \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11433   \ifglsxtrininsertinside\else##2\fi
11434   \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11435 }%
11436 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11437   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11438   \ifglsxtrininsertinside\else##2\fi
11439   \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
11440 }%
11441 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11442   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11443   \ifglsxtrininsertinside\else##2\fi
11444   \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11445 }%
11446 }

```

onguserdesname

```

11447 \newcommand*{\glsxtrshortlonguserdesname}{%
11448   \protect\glsabbrvuserfont{\the\glsshorttok}%
11449   \protect\glsxtruserparen
11450   {\protect\glslonguserfont{\the\glslongpltok}}%
11451   {\the\glslabeltok}%

```

11452 }

long-user-desc Like short-postlong-user but leaves the user to specify the description.

11453 \newabbreviationstyle{short-postlong-user-desc}%

11454 {%

Set accessibility attributes if enabled.

11455 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

Setup the default fields.

11456 \renewcommand*{\CustomAbbreviationFields}{%

11457 name={\glxtrshortlonguserdescname},

11458 sort={\the\glsshorttok},

11459 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%

11460 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

11461 text={\protect\glssabrvuserfont{\the\glsshorttok}},%

11462 plural={\protect\glssabrvuserfont{\the\glsshortpltok}}%

11463 }%

11464 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

11465 \csdef{glxtrpostlink\glscategorylabel}{%

11466 \glxtrifwasfirstuse

11467 {%

11468 \glxtruserparen

11469 {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%

11470 {\glslabel}%

11471 }%

11472 {}%

11473 }%

11474 \glshasattribute{\the\glslabeltok}{regular}%

11475 {%

11476 \glsssetAttribute{\the\glslabeltok}{regular}{false}%

11477 }%

11478 {}%

11479 }%

11480 }%

11481 {%

11482 \GlsXtrUseAbbrStyleFmts{short-postlong-user}%

11483 }

short-user-desc

11484 \newabbreviationstyle{long-short-user-desc}%

11485 {%

Set accessibility attributes if enabled.

11486 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

Setup the default fields.

11487 \renewcommand*{\CustomAbbreviationFields}{%

11488 name={\glsxtrlongshortuserdescname},

11489 sort={\glsxtrlongshortdescsort},%

```

11490 first={\protect\glsfirstlonguserfont{\the\glslongtok}%
11491 \protect\glstruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
11492 {\the\glslabeltok}},%
11493 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
11494 \protect\glstruserparen
11495 {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
11496 text={\protect\glsabbrvfont{\the\glsshorttok}},%
11497 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
11498 }%

```

Unset the regular attribute if it has been set.

```

11499 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11500 \glshasattribute{\the\glslabeltok}{regular}%
11501 {%
11502 \glissetattribute{\the\glslabeltok}{regular}{false}%
11503 }%
11504 {}%
11505 }%
11506 }%
11507 {%
11508 \GlsXtrUseAbbrStyleFmts{long-short-user}%
11509 }

```

short-long-user

```

11510 \newabbreviationstyle{short-long-user}%
11511 {%

```

Set accessibility attributes if enabled.

```

11512 \glstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

\glslonguserfont is used in the description since \glsdesc doesn't set the style. (Now in \glsuserdescription.)

```

11513 \renewcommand*{\CustomAbbreviationFields}{%
11514 name={\glstrshortlongname},
11515 sort={\the\glsshorttok},
11516 description={\protect\glsuserdescription{\the\glslongtok}%
11517 {\the\glslabeltok}},%
11518 first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
11519 \protect\glstruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
11520 {\the\glslabeltok}},%
11521 firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
11522 \protect\glstruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
11523 {\the\glslabeltok}},%

11524 text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11525 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

11526 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

11527 \glshasattribute{\the\glslabeltok}{regular}%
11528 {%
11529 \glissetattribute{\the\glslabeltok}{regular}{false}%
11530 }%
11531 {}%
11532 }%
11533 }%
11534 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11535 \renewcommand*\abbrvpluralsuffix{\glsxtrusersuffix}%
11536 \renewcommand*\glsabbrvfont[1]{\glsabbrvuserfont{##1}}%
11537 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvuserfont{##1}}%
11538 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonguserfont{##1}}%
11539 \renewcommand*\glslongfont[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

11540 \renewcommand*\glsxtrfullformat[2]{%
11541 \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11542 \ifglxtrinsertinside\else##2\fi
11543 \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
11544 }%
11545 \renewcommand*\glsxtrfullplformat[2]{%
11546 \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11547 \ifglxtrinsertinside\else##2\fi
11548 \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11549 }%
11550 \renewcommand*\Glsxtrfullformat[2]{%
11551 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11552 \ifglxtrinsertinside\else##2\fi
11553 \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
11554 }%
11555 \renewcommand*\Glsxtrfullplformat[2]{%
11556 \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11557 \ifglxtrinsertinside\else##2\fi
11558 \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11559 }%
11560 }

```

-long-user-desc

```

11561 \newabbreviationstyle{short-long-user-desc}%
11562 {%

```

Set accessibility attributes if enabled.

```

11563 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

11564 \renewcommand*\CustomAbbreviationFields{%
11565 name={\glsxtrshortlonguserdescname},
11566 sort={\glsxtrshortlongdescsort},%

```

```

11567 first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
11568 \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
11569 {\the\glslabeltok}},%
11570 firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
11571 \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
11572 {\the\glslabeltok}},%
11573 text={\protect\glsabbrvfont{\the\glsshorttok}},%
11574 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
11575 }%

```

Unset the regular attribute if it has been set.

```

11576 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11577 \glshasattribute{\the\glslabeltok}{regular}}%
11578 {%
11579 \glissetattribute{\the\glslabeltok}{regular}{false}}%
11580 }%
11581 {}%
11582 }%
11583 }%
11584 {%
11585 \GlsXtrUseAbbrStyleFmts{short-long-user}}%
11586 }

```

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 `\glsxtrlong` 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.

```

11587 \newrobustcmd*{\glsxtrifhyphenstart}[3]{%
11588 \ifx\glsinsert#1\relax
11589 \expandafter\@glsxtrifhyphenstart#1\relax\relax
11590 \@end@glsxtrifhyphenstart{#2}{#3}}%
11591 \else
11592 \@glsxtrifhyphenstart#1\relax\relax\@end@glsxtrifhyphenstart{#2}{#3}}%
11593 \fi
11594 }

```

`trifhyphenstart`

```

11595 \def\@glsxtrifhyphenstart#1#2\@end@glsxtrifhyphenstart#3#4{%
11596 \ifx-#1\relax#3\else #4\fi
11597 }

```

`longhyphenshort`

```
\glxstrlonghyphenshort{<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.

```
11598 \newcommand*{\glxstrlonghyphenshort}[4]{%
```

Grouping is needed to localise the redefinitions.

```
11599 {%
```

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.

```
11600 \glxstrifhyphenstart{#4}{\def\glxstrwordsep{-}}{}%
11601 \glxstrfirstlonghyphenfont{#2\ifglxstrinsertinside{#4}\fi}%
11602 \ifglxstrinsertinside\else{#4}\fi
11603 \glxstrfullsep{#1}%
11604 \glxstrparen{\glxstrfirstabbrvhyphenfont{#3\ifglxstrinsertinside{#4}\fi}%
11605 \ifglxstrinsertinside\else{#4}\fi}%
11606 }%
11607 }
```

abbrvhyphenfont

```
11608 \newcommand*{\glxabbrvhyphenfont}{\glxabbrvdefaultfont}%
```

abbrvhyphenfont

```
11609 \newcommand*{\glxfirstabbrvhyphenfont}{\glxabbrvhyphenfont}%
```

slonghyphenfont

```
11610 \newcommand*{\glxslonghyphenfont}{\glxslongdefaultfont}%
```

tlonghyphenfont

```
11611 \newcommand*{\glxfirstlonghyphenfont}{\glxslonghyphenfont}%
```

The default short form suffix:

xtrhyphensuffix

```
11612 \newcommand*{\glxstrhyphensuffix}{\glxstrabbrvpluralsuffix}
```

en-short-hyphen Designed for use with the `markwords` attribute.

```
11613 \newabbreviationstyle{long-hyphen-short-hyphen}%
```

```
11614 {%
```

Set accessibility attributes if enabled.

```
11615 \glxstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
11616 \renewcommand*{\CustomAbbreviationFields}{%
```

```
11617 name={\glxstrlongshortname},
```

```
11618 sort={\the\glsshorttok},
```

```

11619 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
11620 \protect\glsxtrfullsep{\the\glslabeltok}%
11621 \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
11622 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
11623 \protect\glsxtrfullsep{\the\glslabeltok}%
11624 \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
11625 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
11626 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
11627 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

11628 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11629 \glsasattribute{\the\glslabeltok}{regular}%
11630 {%
11631 \glssetattribute{\the\glslabeltok}{regular}{false}%
11632 }%
11633 {}%
11634 }%
11635 }%
11636 {%
11637 \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
11638 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
11639 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
11640 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
11641 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

11642 \renewcommand*{\glsxtrfullformat}[2]{%
11643 \glsxtrlonghyphenshort{##1}{\glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
11644 }%
11645 \renewcommand*{\glsxtrfullplformat}[2]{%
11646 \glsxtrlonghyphenshort{##1}{\glsaccesslongpl{##1}}%
11647 {\glsaccessshortpl{##1}}{##2}%
11648 }%
11649 \renewcommand*{\Glsxtrfullformat}[2]{%
11650 \glsxtrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
11651 }%
11652 \renewcommand*{\Glsxtrfullplformat}[2]{%
11653 \glsxtrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
11654 {\glsaccessshortpl{##1}}{##2}%
11655 }%
11656 }

```

`ort-hyphen-desc` Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

11657 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
11658 {%

```

Set accessibility attributes if enabled.

```

11659 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.


```

11660 \renewcommand*{\CustomAbbreviationFields}{%
11661     name={\glxtrlongshortdescname},
11662     sort={\glxtrlongshortdescsort},
11663     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
11664         \protect\glxtrfullsep{\the\glslabeltok}%
11665         \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
11666     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
11667         \protect\glxtrfullsep{\the\glslabeltok}%
11668         \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
11669     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
11670     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
11671 }%

```

Unset the regular attribute if it has been set.

```

11672 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11673     \glshasattribute{\the\glslabeltok}{regular}%
11674     {%
11675         \glissetattribute{\the\glslabeltok}{regular}{false}%
11676     }%
11677     {}%
11678 }%
11679 }%
11680 {%
11681     \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
11682 }

```

nghyphennoshort

`\glxtrlonghyphennoshort{<label>}{<long>}{<insert>}`

```

11683 \newcommand*{\glxtrlonghyphennoshort}[3]{%

```

Grouping is needed to localise the redefinitions.

```

11684 {%

```

If *<insert>* starts with a hyphen, redefine `\glxtrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.) No change is made to `\glxtrwordsep` if *<insert>* doesn't start with a hyphen.

```

11685     \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{%
11686     \glsfirstlonghyphenfont{#2\ifglxtrininsertinside{#3}\fi}%
11687     \ifglxtrininsertinside\else{#3}\fi
11688 }%
11689 }

```

hort-desc-noreg

This version doesn't show the short form (except explicitly with `\glxtrshort`). Since `\glxtrshort` 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.

```

11690 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
11691 {%
11692   \renewcommand*{\CustomAbbreviationFields}{%
11693     name={\glxstrlongnoshortdescname},
11694     sort={\expandonce\glxstrorglong},
11695     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
11696     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
11697     text={\protect\glslonghyphenfont{\the\glslongtok}},%
11698     plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
11699   }%

```

Unset the regular attribute if it has been set.

```

11700 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11701   \glshasattribute{\the\glslabeltok}{regular}%
11702   {%
11703     \glissetattribute{\the\glslabeltok}{regular}{false}%
11704   }%
11705   {}}%
11706 }%
11707 }%
11708 {%
11709   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11710 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
11711 \renewcommand*{\glssabrvfont}[1]{\glssabrvdefaultfont{##1}}%
11712 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
11713 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
11714 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

11715 \renewcommand*{\glxtrsubsequentfmt}[2]{%
11716   \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
11717 }%
11718 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
11719   \glxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
11720 }%
11721 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
11722   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
11723 }%
11724 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
11725   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
11726 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

11727 \renewcommand*{\glxtrinilinefullformat}[2]{%
11728   \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
11729   \glxtrfullsep{##1}%
11730   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshort{##1}}}%
11731 }%
11732 \renewcommand*{\glxtrinilinefullplformat}[2]{%

```

```

11733 \glxstrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
11734 \glxstrfullsep{##1}%
11735 \glxstrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
11736 }%
11737 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11738 \glxstrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
11739 \glxstrfullsep{##1}%
11740 \glxstrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
11741 }%
11742 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11743 \glxstrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
11744 \glxstrfullsep{##1}%
11745 \glxstrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
11746 }%

```

The first use full form only displays the long form.

```

11747 \renewcommand*{\glxstrfullformat}[2]{%
11748 \glxstrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
11749 }%
11750 \renewcommand*{\glxstrfullplformat}[2]{%
11751 \glxstrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
11752 }%
11753 \renewcommand*{\Glsxstrfullformat}[2]{%
11754 \glxstrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
11755 }%
11756 \renewcommand*{\Glsxstrfullplformat}[2]{%
11757 \glxstrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
11758 }%
11759 }

```

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.

```

11760 \newabbreviationstyle{long-hyphen-noshort-noreg}%
11761 {%

```

Set accessibility attributes if enabled.

```

11762 \glxstrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```

11763 \renewcommand*{\CustomAbbreviationFields}{%
11764 name={\glxstrlongnoshortname},
11765 sort={\the\glsshorttok},
11766 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
11767 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
11768 text={\protect\glslonghyphenfont{\the\glslongtok}},%
11769 plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
11770 description={\the\glslongtok}%
11771 }%

```

Unset the regular attribute if it has been set.

```
11772 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11773   \glshasattribute{\the\glslabeltok}{regular}%
11774   {%
11775     \glsselattribute{\the\glslabeltok}{regular}{false}%
11776   }%
11777   {}%
11778 }%
11779}%
11780{%
```



```
11781 \GlsXtrUseAbbrStyleFmts{long-hyphen-noshort-desc-noreg}%
11782 }
```

lsxtrlonghyphen

`\lsxtrlonghyphen{<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.

```
11783 \newcommand*{\lsxtrlonghyphen}[3]{%
```

Grouping is needed to localise the redefinitions.

```
11784   {%
11785     \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{}%
11786     \glsfirslonghyphenfont{#1}%
11787   }%
11788 }
```

posthyphenshort

`\lsxtrposthyphenshort{<label>}{<insert>}`

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like `\lsxtrlonghyphenshort` but omits the `<long>` part. This always uses the singular short form.

```
11789 \newcommand*{\lsxtrposthyphenshort}[2]{%
11790   {%
11791     \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
11792     \ifglxtrininsertinside{\glsfirslonghyphenfont{#2}}\else{#2}\fi
11793     \glxtrfullsep{#1}%
11794     \glxtrparen
11795     {\glsfirstabbrvhyphenfont{\glsentryshort{#1}}\ifglxtrininsertinside{#2}\fi}%
11796     \ifglxtrininsertinside\else{#2}\fi
11797   }%
11798 }%
11799 }
```

ypkensubsequent

`\glxtrposthyphensubsequent{<label>}{<insert>}`

Format in the post-link hook for subsequent use. The label is ignored by default.

```
11800 \newcommand*{\glxtrposthyphensubsequent}[2]{%
11801   \glssabrvfont{\ifglxtrininsertinside {#2}\fi}%
11802   \ifglxtrininsertinside \else{#2}\fi
11803 }
```

ostshort-hyphen Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```
11804 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
11805 {%
```

Set accessibility attributes if enabled.

```
11806   \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
11807   \renewcommand*{\CustomAbbreviationFields}{%
11808     name={\glxtrlongshortname},
11809     sort={\the\glsshorttok},
11810     first={\protect\glslonghyphenfont{\the\glslongtok}},%
11811     firstplural={\protect\glslonghyphenfont{\the\glslongpltok}},%
11812     text={\protect\glssabrvhyphenfont{\the\glsshorttok}},%
11813     plural={\protect\glssabrvhyphenfont{\the\glsshortpltok}},%
11814     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
11815   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11816     \csdef{glxtrpostlink\glscategorylabel}{%
11817       \glxtrifwasfirstuse
11818       {%
11819         \glxtrposthyphenshort{\glslabel}{\glssinsert}%
11820       }%
11821     }%
```

Put the insertion into the post-link:

```
11822       \glxtrposthyphensubsequent{\glslabel}{\glssinsert}%
11823     }%
11824   }%
11825   \glshasattribute{\the\glslabeltok}{regular}%
11826   {%
11827     \glissetattribute{\the\glslabeltok}{regular}{false}%
11828   }%
11829   {}%
11830 }%
11831 }%
11832 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
11833   \renewcommand*{\abrvpluralsuffix}{\glxtrabrvpluralsuffix}%
```

```

11834 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
11835 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
11836 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
11837 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

11838 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
11839   \glsabbrvfont{\glsaccessshort{##1}}}%
11840 }%
11841 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
11842   \glsabbrvfont{\glsaccessshortpl{##1}}}%
11843 }%
11844 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
11845   \glsabbrvfont{\Glsaccessshort{##1}}}%
11846 }%
11847 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
11848   \glsabbrvfont{\Glsaccessshortpl{##1}}}%
11849 }%

```

First use full form:

```

11850 \renewcommand*{\glsxtrfullformat}[2]{%
11851   \glsxtrlonghyphen{\glsaccesslong{##1}}{##1}{##2}%
11852 }%
11853 \renewcommand*{\glsxtrfullplformat}[2]{%
11854   \glsxtrlonghyphen{\glsaccesslongpl{##1}}{##1}{##2}%
11855 }%
11856 \renewcommand*{\Glsxtrfullformat}[2]{%
11857   \glsxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
11858 }%
11859 \renewcommand*{\Glsxtrfullplformat}[2]{%
11860   \glsxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
11861 }%

```

In-line format.

```

11862 \renewcommand*{\glsxtrinlinefullformat}[2]{%
11863   \glsfirstlonghyphenfont{\glsaccesslong{##1}}%
11864   \ifglsxtrininsertinside{##2}\fi}%
11865   \ifglsxtrininsertinside \else{##2}\fi
11866 }%
11867 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
11868   \glsfirstlonghyphenfont{\glsaccesslongpl{##1}}%
11869   \ifglsxtrininsertinside{##2}\fi}%
11870   \ifglsxtrininsertinside \else{##2}\fi
11871 }%
11872 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11873   \glsfirstlonghyphenfont{\Glsaccesslong{##1}}%
11874   \ifglsxtrininsertinside{##2}\fi}%
11875   \ifglsxtrininsertinside \else{##2}\fi
11876 }%
11877 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11878   \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}}%

```

```

11879     \ifglxtrinsertinside{##2}\fi}%
11880     \ifglxtrinsertinside \else{##2}\fi
11881 }%
11882 }

```

ort-hyphen-desc Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

11883 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
11884 {%

```

Set accessibility attributes if enabled.

```

11885 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

11886 \renewcommand*{\CustomAbbreviationFields}{%
11887     name={\glxtrlongshortdescname},
11888     sort={\glxtrlongshortdescsort},%
11889     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
11890     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
11891     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
11892     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
11893 }%
11894 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11895     \csdef{glxtrpostlink\glscategorylabel}{%
11896         \glxtrifwasfirstuse
11897         {%
11898             \glxtrposthyphenshort{\glslabel}{\glsinsert}%
11899         }%
11900     }%

```

Put the insertion into the post-link:

```

11901         \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
11902     }%
11903 }%
11904 \glshasattribute{\the\glslabeltok}{regular}%
11905 {%
11906     \glissetattribute{\the\glslabeltok}{regular}{false}%
11907 }%
11908 {}%
11909 }%
11910 }%
11911 {%
11912     \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
11913 }

```

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.

11914 \newcommand*{\glxtrshorthyphenlong}[4]{%

Grouping is needed to localise the redefinitions.

11915 {%

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.)

11916 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{}%

11917 \glsfirstabbrvhyphenfont{#2\ifglxtrininsertinside{#4}\fi}%

11918 \ifglxtrininsertinside\else{#4}\fi

11919 \glxtrfullsep{#1}%

11920 \glxtrparen{\glsfirstlonghyphenfont{#3\ifglxtrininsertinside{#4}\fi}%

11921 \ifglxtrininsertinside\else{#4}\fi}%

11922 }%

11923 }

hen-long-hyphen Designed for use with the markwords attribute.

11924 \newabbreviationstyle{short-hyphen-long-hyphen}%

11925 {%

Set accessibility attributes if enabled.

11926 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

Setup the default fields.

11927 \renewcommand*{\CustomAbbreviationFields}{%

11928 name={\glxtrshortlongname},

11929 sort={\the\glsshorttok},

11930 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%

11931 \protect\glxtrfullsep{\the\glslabeltok}%

11932 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%

11933 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%

11934 \protect\glxtrfullsep{\the\glslabeltok}%

11935 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%

11936 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%

11937 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%

11938 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

Unset the regular attribute if it has been set.

11939 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

11940 \glshasattribute{\the\glslabeltok}{regular}%

11941 {%

11942 \glssetattribute{\the\glslabeltok}{regular}{false}%

11943 }%

11944 {}%

11945 }%

11946 }%

11947 {%

11948 \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%

11949 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%

11950 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%


```

11951 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
11952 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

11953 \renewcommand*{\glsxtrfullformat}[2]{%
11954   \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
11955 }%
11956 \renewcommand*{\glsxtrfullplformat}[2]{%
11957   \glsxtrshorthyphenlong{##1}%
11958   {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
11959 }%
11960 \renewcommand*{\Glsxtrfullformat}[2]{%
11961   \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
11962 }%
11963 \renewcommand*{\Glsxtrfullplformat}[2]{%
11964   \glsxtrshorthyphenlong{##1}%
11965   {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
11966 }%
11967 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

11968 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
11969 {%

```

Set accessibility attributes if enabled.

```

11970 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

11971 \renewcommand*{\CustomAbbreviationFields}{%
11972   name={\glsxtrshortlongdescname},
11973   sort={\glsxtrshortlongdescsort},
11974   first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%
11975   \protect\glsxtrfullsep{\the\glslabeltok}}%
11976   \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
11977   firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%
11978   \protect\glsxtrfullsep{\the\glslabeltok}}%
11979   \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
11980   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}}},%
11981   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
11982 }%

```

Unset the regular attribute if it has been set.

```

11983 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11984   \glsasattribute{\the\glslabeltok}{regular}}%
11985   {%
11986     \glssetattribute{\the\glslabeltok}{regular}{false}}%
11987   }%
11988   {}%
11989 }%
11990 }%
11991 {%

```

```

11992 \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
11993 }

```

sxtrshorthyphen

```
\glsxtrshorthyphen{<short>}{<label>}{<insert>}
```

Used by short-hyphen-postlong-hyphen. The *<insert>* is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```
11994 \newcommand*{\glsxtrshorthyphen}[3]{%
```

Grouping is needed to localise the redefinitions.

```

11995 {%
11996   \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
11997   \glsfirstabbrvhyphenfont{#1}%
11998 }%
11999 }

```

rpostthyphenlong

```
\glsxtrpostthyphenlong{<label>}{<insert>}
```

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glsxtrshorthyphenlong` but omits the *<short>* part. This always uses the singular long form.

```

12000 \newcommand*{\glsxtrpostthyphenlong}[2]{%
12001 {%
12002   \glsxtrifhyphenstart{#2}{\def\glsxtrwordsep{-}}{}%
12003   \ifglsxtrininsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
12004   \glsxtrfullsep{#1}%
12005   \glsxtrparen
12006   {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglsxtrininsertinside{#2}\fi}%
12007   \ifglsxtrininsertinside\else{#2}\fi
12008 }%
12009 }%
12010 }

```

postlong-hyphen Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

12011 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
12012 {%

```

Set accessibility attributes if enabled.

```
12013 \glsxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```

12014 \renewcommand*{\CustomAbbreviationFields}{%
12015   name={\glsxtrshortlongname},

```

```

12016     sort={\the\glsshorttok},
12017     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
12018     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
12019     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
12020     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
12021     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
12022 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12023   \csdef{glsxtrpostlink\glscategorylabel}{%
12024     \glsxtrifwasfirstuse
12025     {%
12026       \glsxtrposthyphenlong{\glslabel}{\glsinsert}%
12027     }%
12028   }%

```

Put the insertion into the post-link:

```

12029     \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
12030   }%
12031 }%
12032 \glsattribute{\the\glslabeltok}{regular}%
12033 {%
12034   \glssetAttribute{\the\glslabeltok}{regular}{false}%
12035 }%
12036 {}%
12037 }%
12038 }%
12039 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

12040 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
12041 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
12042 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
12043 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
12044 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

12045 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
12046   \glsabbrvfont{\glsaccessshort{##1}}%
12047 }%
12048 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
12049   \glsabbrvfont{\glsaccessshortpl{##1}}%
12050 }%
12051 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
12052   \glsabbrvfont{\Glsaccessshort{##1}}%
12053 }%
12054 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
12055   \glsabbrvfont{\Glsaccessshortpl{##1}}%
12056 }%

```

First use full form:

```

12057 \renewcommand*{\glsxtrfullformat}[2]{%
12058   \glsxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%

```

```

12059 }%
12060 \renewcommand*{\glxtrfullplformat}[2]{%
12061   \glxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
12062 }%
12063 \renewcommand*{\Glsxtrfullformat}[2]{%
12064   \glxtrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
12065 }%
12066 \renewcommand*{\Glsxtrfullplformat}[2]{%
12067   \glxtrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
12068 }%

```

In-line format. Commands like `\glxtrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

12069 \renewcommand*{\glxtrinlinefullformat}[2]{%
12070   \glsfirstabbrvhyphenfont{\glsaccessshort{##1}%
12071     \ifglxtrininsertinside{##2}\fi}%
12072   \ifglxtrininsertinside \else{##2}\fi
12073 }%
12074 \renewcommand*{\glxtrinlinefullplformat}[2]{%
12075   \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}%
12076     \ifglxtrininsertinside{##2}\fi}%
12077   \ifglxtrininsertinside \else{##2}\fi
12078 }%
12079 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
12080   \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}%
12081     \ifglxtrininsertinside{##2}\fi}%
12082   \ifglxtrininsertinside \else{##2}\fi
12083 }%
12084 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
12085   \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}%
12086     \ifglxtrininsertinside{##2}\fi}%
12087   \ifglxtrininsertinside \else{##2}\fi
12088 }%
12089 }

```

`ong-hyphen-desc` Like `short-hyphen-postlong-hyphen` but the description must be supplied by the user.

```

12090 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
12091 {%

```

Set accessibility attributes if enabled.

```

12092   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

12093   \renewcommand*{\CustomAbbreviationFields}{%
12094     name={\glxtrshortlongdescname},
12095     sort={\glxtrshortlongdescsort},%
12096     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
12097     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
12098     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
12099     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%

```

```

12100 }%
12101 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12102   \csdef{glxtrpostlink\glscategorylabel}{%
12103     \glxtrifwasfirstuse
12104     {%
12105       \glxtrposthyphenlong{\glslabel}{\glinsert}%
12106     }%
12107     {%

```

Put the insertion into the post-link:

```

12108       \glxtrposthyphensubsequent{\glslabel}{\glinsert}%
12109     }%
12110   }%
12111   \glshasattribute{\the\glslabeltok}{regular}%
12112   {%
12113     \glissetattribute{\the\glslabeltok}{regular}{false}%
12114   }%
12115   {}%
12116 }%
12117 }%
12118 {%
12119   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
12120 }

```

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

```
12121 \newcommand*{\glabbrvonlyfont}{\glabbrvdefaultfont}%
```

stabbrvonlyfont

```
12122 \newcommand*{\glfirstabbrvonlyfont}{\glabbrvonlyfont}%
```

glslongonlyfont

```
12123 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%
```

rstlongonlyfont

```
12124 \newcommand*{\glfirstlongonlyfont}{\glslongonlyfont}%
```

The default short form suffix:

lsxtronlysuffix

```
12125 \newcommand*{\glxtronlysuffix}{\glxtrabbrvpluralsuffix}
```

\glxtronlyname The default name format for this style.

```

12126 \newcommand*{\glxtronlyname}{%
12127   \protect\glabbrvonlyfont{\the\glsshorttok}%
12128 }

```

only-short-only

```
12129 \newabbreviationstyle{long-only-short-only}%
12130 {%
```

Set accessibility attributes if enabled.

```
12131 \glstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
12132 \renewcommand*{\CustomAbbreviationFields}{%
12133   name={\glstronlyname},
12134   sort={\the\glsshorttok},
12135   first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
12136   firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
12137   text={\protect\glssabrvonlyfont{\the\glsshorttok}},%
12138   plural={\protect\glssabrvonlyfont{\the\glsshortpltok}},%
12139   description={\protect\glslongonlyfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
12140 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12141   \glshasattribute{\the\glslabeltok}{regular}%
12142   {%
12143     \glissetattribute{\the\glslabeltok}{regular}{false}%
12144   }%
12145   {}%
12146 }%
12147 }%
12148 {%
```

```
12149 \renewcommand*{\abrvpluralsuffix}{\glstronlysuffix}%
12150 \renewcommand*{\glssabrvfont}[1]{\glssabrvonlyfont{##1}}%
12151 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvonlyfont{##1}}%
12152 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
12153 \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%
```

The first use full form doesn't show the short form.

```
12154 \renewcommand*{\glstrfullformat}[2]{%
12155   \glsfirstlongonlyfont{\glssaccesslong{##1}\ifglstrinsertinside##2\fi}%
12156   \ifglstrinsertinside\else##2\fi
12157 }%
12158 \renewcommand*{\glstrfullplformat}[2]{%
12159   \glsfirstlongonlyfont{\glssaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
12160   \ifglstrinsertinside\else##2\fi
12161 }%
12162 \renewcommand*{\Glsxtrfullformat}[2]{%
12163   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglstrinsertinside##2\fi}%
12164   \ifglstrinsertinside\else##2\fi
12165 }%
12166 \renewcommand*{\Glsxtrfullplformat}[2]{%
12167   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
12168   \ifglstrinsertinside\else##2\fi
12169 }%
```

The inline full form does show the short form.

```

12170 \renewcommand*{\glxtrinlinefullformat}[2]{%
12171   \glsfirslongonlyfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
12172   \ifglxtrinsertinside\else##2\fi
12173   \glxtrfullsep{##1}%
12174   \glxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
12175 }%
12176 \renewcommand*{\glxtrinlinefullplformat}[2]{%
12177   \glsfirslongonlyfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
12178   \ifglxtrinsertinside\else##2\fi
12179   \glxtrfullsep{##1}%
12180   \glxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
12181 }%
12182 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
12183   \glsfirslongonlyfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
12184   \ifglxtrinsertinside\else##2\fi
12185   \glxtrfullsep{##1}%
12186   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
12187 }%
12188 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
12189   \glsfirslongonlyfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
12190   \ifglxtrinsertinside\else##2\fi
12191   \glxtrfullsep{##1}%
12192   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
12193 }%
12194 }

```

xtronlydescsort

```

12195 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

12196 \newcommand*{\glxtronlydescname}{%
12197   \protect\glslongfont{\the\glslongtok}}%
12198 }

```

short-only-desc

```

12199 \newabbreviationstyle{long-only-short-only-desc}%
12200 {}%

```

Set accessibility attributes if enabled.

```

12201 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

12202 \renewcommand*{\CustomAbbreviationFields}{%
12203   name={\glxtronlydescname},
12204   sort={\glxtronlydescsort},%
12205   first={\protect\glsfirslongonlyfont{\the\glslongtok}},%
12206   firstplural={\protect\glsfirslongonlyfont{\the\glslongpltok}},%
12207   text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%

```

```

12208 plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}}%
12209 }%

Unset the regular attribute if it has been set.
12210 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12211 \glshasattribute{\the\glslabeltok}{regular}%
12212 {%
12213 \glssetAttribute{\the\glslabeltok}{regular}{false}%
12214 }%
12215 {}%
12216 }%
12217}%
12218{%
12219 \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
12220}

```

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 `\glsentryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glsabbrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref's \texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glsxtrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to `false`, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that glossaries automatically loads `textcase`, so the label can be protected from case change with `textcase's \NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
12221 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```

12222 \renewcommand*{\markright}[1]{%
12223 \glsxtrmarkhook

```



```

12224 \@glsxtr@org@markright{\@glsxtrinmark#1\@glsxtrnotinmark}%
12225 \glsxtrrestoremarkhook
12226 }

```

\markboth Save original definition:

```
12227 \let\@glsxtr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```

12228 \renewcommand*{\markboth}[2]{%
12229 \glsxtrmarkhook
12230 \@glsxtr@org@markboth
12231 {\@glsxtrinmark#1\@glsxtrnotinmark}%
12232 {\@glsxtrinmark#2\@glsxtrnotinmark}%
12233 \glsxtrrestoremarkhook
12234 }

```

Also do this for \@starttoc

\@starttoc Save original definition:

```
12235 \let\@glsxtr@org@@starttoc\@starttoc
```

Redefine:

```

12236 \renewcommand*{\@starttoc}[1]{%
12237 \glsxtrmarkhook
12238 \@glsxtrinmark
12239 \@glsxtr@org@@starttoc{#1}%
12240 \@glsxtrnotinmark
12241 \glsxtrrestoremarkhook
12242 }

```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```

12243 \newcommand*{\glsxtrRevertMarks}{%
12244 \let\markright\@glsxtr@org@markright
12245 \let\markboth\@glsxtr@org@markboth
12246 \let\@starttoc\@glsxtr@org@@starttoc
12247 }

```

rRevertTocMarks Just restores \@starttoc.

```

12248 \newcommand*{\glsxtrRevertTocMarks}{%
12249 \let\@starttoc\@glsxtr@org@@starttoc
12250 }

```

\glsxtrifinmark

```
12251 \newcommand*{\glsxtrifinmark}[2]{#2}
```

\@glsxtrinmark

```

12252 \newrobustcmd*{\@glsxtrinmark}{%
12253 \let\glsxtrifinmark\@firstoftwo
12254 }

```

glsxtrnotinmark

```
12255 \newrobustcmd*{\@glsxtrnotinmark}{%
12256   \let\glsxtrifinmark\@secondoftwo
12257 }
```

eorpdforheading

```
12258 \ifdef\teorpdfstring
12259 {
12260   \newcommand*{\glsxtrtitleorpdforheading}[3]{\teorpdfstring{#1}{#2}}
12261 }
12262 {
12263   \newcommand*{\glsxtrtitleorpdforheading}[3]{#1}
12264 }
```

\glsxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```
12265 \newcommand*{\glsxtrmarkhook}{%
```

Save current definitions:

```
12266 \let\@glsxtr@org@MakeUppercase\MakeUppercase
12267 \let\@glsxtr@org@glsxtrtitleorpdforheading\glsxtrtitleorpdforheading
12268 \let\@glsxtr@org@glsxtrtitleshort\glsxtrtitleshort
12269 \let\@glsxtr@org@glsxtrtitleshortpl\glsxtrtitleshortpl
12270 \let\@glsxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
12271 \let\@glsxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
12272 \let\@glsxtr@org@glsxtrtitlename\glsxtrtitlename
12273 \let\@glsxtr@org@Glsxtrtitlename\Glsxtrtitlename
12274 \let\@glsxtr@org@glsxtrtitletext\glsxtrtitletext
12275 \let\@glsxtr@org@Glsxtrtitletext\Glsxtrtitletext
12276 \let\@glsxtr@org@glsxtrtitleplural\glsxtrtitleplural
12277 \let\@glsxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
12278 \let\@glsxtr@org@glsxtrtitlefirst\glsxtrtitlefirst
12279 \let\@glsxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
12280 \let\@glsxtr@org@glsxtrtitlefirstplural\glsxtrtitlefirstplural
12281 \let\@glsxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
12282 \let\@glsxtr@org@glsxtrtitlelong\glsxtrtitlelong
12283 \let\@glsxtr@org@glsxtrtitlelongpl\glsxtrtitlelongpl
12284 \let\@glsxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
12285 \let\@glsxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
12286 \let\@glsxtr@org@glsxtrtitlefull\glsxtrtitlefull
12287 \let\@glsxtr@org@glsxtrtitlefullpl\glsxtrtitlefullpl
12288 \let\@glsxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
12289 \let\@glsxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl
```

New definitions

```
12290 \let\glsxtrifinmark\@firstoftwo
12291 \let\MakeUppercase\MakeTextUppercase
12292 \let\glsxtrtitleorpdforheading\@thirdofthree
12293 \let\glsxtrtitleshort\glsxtrheadshort
12294 \let\glsxtrtitleshortpl\glsxtrheadshortpl
```

```

12295 \let\Glsxtrtitleshort\Glsxtrheadshort
12296 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
12297 \let\Glsxtrtitlename\Glsxtrheadname
12298 \let\Glsxtrtitlename\Glsxtrheadname
12299 \let\Glsxtrtitletext\Glsxtrheadtext
12300 \let\Glsxtrtitletext\Glsxtrheadtext
12301 \let\Glsxtrtitleplural\Glsxtrheadplural
12302 \let\Glsxtrtitleplural\Glsxtrheadplural
12303 \let\Glsxtrtitlefirst\Glsxtrheadfirst
12304 \let\Glsxtrtitlefirst\Glsxtrheadfirst
12305 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
12306 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
12307 \let\Glsxtrtitlelong\Glsxtrheadlong
12308 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
12309 \let\Glsxtrtitlelong\Glsxtrheadlong
12310 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
12311 \let\Glsxtrtitlefull\Glsxtrheadfull
12312 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
12313 \let\Glsxtrtitlefull\Glsxtrheadfull
12314 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
12315 }

```

`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.)

```

12316 \newcommand*{\Glsxtrrestoremarkhook}{%
12317   \let\Glsxtrifinmark\@secondoftwo
12318   \let\MakeUppercase\@Glsxtr@org@MakeUppercase
12319   \let\Glsxtrtitleorpdforheading\@Glsxtr@org@Glsxtrtitleorpdforheading
12320   \let\Glsxtrtitleshort\@Glsxtr@org@Glsxtrtitleshort
12321   \let\Glsxtrtitleshortpl\@Glsxtr@org@Glsxtrtitleshortpl
12322   \let\Glsxtrtitleshort\@Glsxtr@org@Glsxtrtitleshort
12323   \let\Glsxtrtitleshortpl\@Glsxtr@org@Glsxtrtitleshortpl
12324   \let\Glsxtrtitlename\@Glsxtr@org@Glsxtrtitlename
12325   \let\Glsxtrtitlename\@Glsxtr@org@Glsxtrtitlename
12326   \let\Glsxtrtitletext\@Glsxtr@org@Glsxtrtitletext
12327   \let\Glsxtrtitletext\@Glsxtr@org@Glsxtrtitletext
12328   \let\Glsxtrtitleplural\@Glsxtr@org@Glsxtrtitleplural
12329   \let\Glsxtrtitleplural\@Glsxtr@org@Glsxtrtitleplural
12330   \let\Glsxtrtitlefirst\@Glsxtr@org@Glsxtrtitlefirst
12331   \let\Glsxtrtitlefirst\@Glsxtr@org@Glsxtrtitlefirst
12332   \let\Glsxtrtitlefirstplural\@Glsxtr@org@Glsxtrtitlefirstplural
12333   \let\Glsxtrtitlefirstplural\@Glsxtr@org@Glsxtrtitlefirstplural
12334   \let\Glsxtrtitlelong\@Glsxtr@org@Glsxtrtitlelong
12335   \let\Glsxtrtitlelongpl\@Glsxtr@org@Glsxtrtitlelongpl
12336   \let\Glsxtrtitlelong\@Glsxtr@org@Glsxtrtitlelong
12337   \let\Glsxtrtitlelongpl\@Glsxtr@org@Glsxtrtitlelongpl
12338   \let\Glsxtrtitlefull\@Glsxtr@org@Glsxtrtitlefull

```

```

12339 \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
12340 \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
12341 \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl
12342 }

```

Instead of using one document-wide conditional, use `headuc` attribute to determine whether or not to use the all upper case form.

glsxtrheadshort Command used to display short form in the page header.

```

12343 \newcommand*{\glsxtrheadshort}[1]{%
12344 \protect\NoCaseChange
12345 {%
12346 \glsifattribute{#1}{headuc}{true}%
12347 {%
12348 \GLSxtrshort[noindex,hyper=false]{#1}[]%
12349 }%
12350 {%
12351 \glsxtrshort[noindex,hyper=false]{#1}[]%
12352 }%
12353 }%
12354 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

12355 \newrobustcmd*{\lsxtrtitleshort}[1]{%
12356 \glsxtrshort[noindex,hyper=false]{#1}[]%
12357 }

```

sxtrheadshortpl Command used to display plural short form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrshortpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

12358 \newcommand*{\glsxtrheadshortpl}[1]{%
12359 \protect\NoCaseChange
12360 {%
12361 \glsifattribute{#1}{headuc}{true}%
12362 {%
12363 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
12364 }%
12365 {%
12366 \glsxtrshortpl[noindex,hyper=false]{#1}[]%
12367 }%
12368 }%
12369 }

```

lsxtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```

12370 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
12371 \glsxtrshortpl[noindex,hyper=false]{#1}[]%
12372 }

```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```

12373 \newcommand*{\Glsxtrheadshort}[1]{%
12374   \protect\NoCaseChange
12375   {%
12376     \glsifattribute{#1}{headuc}{true}%
12377     {%
12378       \Glsxtrshort[noindex,hyper=false]{#1}[]%
12379     }%
12380     {%
12381       \Glsxtrshort[noindex,hyper=false]{#1}[]%
12382     }%
12383   }%
12384 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12385 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
12386   \Glsxtrshort[noindex,hyper=false]{#1}[]%
12387 }

```

LSxtrtitleshort Command to display short form of abbreviation in section title and table of contents in all upper case.

```

12388 \newrobustcmd*{\GLSxtrtitleshort}[1]{%
12389   \Glsxtrshort[noindex,hyper=false]{#1}[]%
12390 }

```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```

12391 \newcommand*{\Glsxtrheadshortpl}[1]{%
12392   \protect\NoCaseChange
12393   {%
12394     \glsifattribute{#1}{headuc}{true}%
12395     {%
12396       \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
12397     }%
12398     {%
12399       \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
12400     }%
12401   }%
12402 }

```

lsxtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12403 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
12404   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
12405 }

```

`\xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents in all upper case.

```
12406 \newrobustcmd*{\GLSxtrtitleshortpl}[1]{%
12407   \GLSxtrshortpl [noindex,hyper=false]{#1} []%
12408 }
```

`\glxtrheadname` As above but for the name value.

```
12409 \newcommand*{\glxtrheadname}[1]{%
12410   \protect\NoCaseChange
12411   {%
12412     \glsifattribute{#1}{headuc}{true}%
12413     {%
12414       \GLSname [noindex,hyper=false]{#1} []%
12415     }%
12416     {%
12417       \glsname [noindex,hyper=false]{#1} []%
12418     }%
12419   }%
12420 }
```

`\glxtrtitlename` Command to display name value in section title and table of contents.

```
12421 \newrobustcmd*{\glxtrtitlename}[1]{%
12422   \glsname [noindex,hyper=false]{#1} []%
12423 }
```

`\Glsxtrheadname` First letter converted to upper case

```
12424 \newcommand*{\Glsxtrheadname}[1]{%
12425   \protect\NoCaseChange
12426   {%
12427     \glsifattribute{#1}{headuc}{true}%
12428     {%
12429       \GLSname [noindex,hyper=false]{#1} []%
12430     }%
12431     {%
12432       \Glsname [noindex,hyper=false]{#1} []%
12433     }%
12434   }%
12435 }
```

`\Glsxtrtitlename` Command to display name value in section title and table of contents with the first letter changed to upper case.

```
12436 \newrobustcmd*{\Glsxtrtitlename}[1]{%
12437   \Glsname [noindex,hyper=false]{#1} []%
12438 }
```

`\GLSxtrtitlename` Command to display name value in section title and table of contents in all upper case.

```
12439 \newrobustcmd*{\GLSxtrtitlename}[1]{%
12440   \GLSname [noindex,hyper=false]{#1} []%
12441 }
```

`\glsxtrheadtext` As above but for the text value.

```
12442 \newcommand*{\glsxtrheadtext}[1]{%
12443   \protect\NoCaseChange
12444   {%
12445     \glsifattribute{#1}{headuc}{true}%
12446     {%
12447       \GLStext[noindex,hyper=false]{#1}[]%
12448     }%
12449     {%
12450       \glstext[noindex,hyper=false]{#1}[]%
12451     }%
12452   }%
12453 }
```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```
12454 \newrobustcmd*{\glsxtrtitletext}[1]{%
12455   \glstext[noindex,hyper=false]{#1}[]%
12456 }
```

`\Glsxtrheadtext` First letter converted to upper case

```
12457 \newcommand*{\Glsxtrheadtext}[1]{%
12458   \protect\NoCaseChange
12459   {%
12460     \glsifattribute{#1}{headuc}{true}%
12461     {%
12462       \GLStext[noindex,hyper=false]{#1}[]%
12463     }%
12464     {%
12465       \Glstext[noindex,hyper=false]{#1}[]%
12466     }%
12467   }%
12468 }
```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```
12469 \newrobustcmd*{\Glsxtrtitletext}[1]{%
12470   \Glstext[noindex,hyper=false]{#1}[]%
12471 }
```

`GLSxtrtitletext` Command to display text value in section title and table of contents in all upper case.

```
12472 \newrobustcmd*{\GLSxtrtitletext}[1]{%
12473   \GLStext[noindex,hyper=false]{#1}[]%
12474 }
```

`lsxtrheadplural` As above but for the plural value.

```
12475 \newcommand*{\lsxtrheadplural}[1]{%
12476   \protect\NoCaseChange
12477   {%
```

```

12478 \glsifattribute{#1}{headuc}{true}%
12479 {%
12480 \GLSplural[noindex,hyper=false]{#1}[]%
12481 }%
12482 {%
12483 \glsplural[noindex,hyper=false]{#1}[]%
12484 }%
12485 }%
12486 }

```

sxtrtitleplural Command to display plural value in section title and table of contents.

```

12487 \newrobustcmd*{\glsxtrtitleplural}[1]{%
12488 \glsplural[noindex,hyper=false]{#1}[]%
12489 }

```

lsxtrheadplural Convert first letter to upper case.

```

12490 \newcommand*{\Glsxtrheadplural}[1]{%
12491 \protect\NoCaseChange
12492 {%
12493 \glsifattribute{#1}{headuc}{true}%
12494 {%
12495 \GLSplural[noindex,hyper=false]{#1}[]%
12496 }%
12497 {%
12498 \Glsplural[noindex,hyper=false]{#1}[]%
12499 }%
12500 }%
12501 }

```

sxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```

12502 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
12503 \Glsplural[noindex,hyper=false]{#1}[]%
12504 }

```

Sxtrtitleplural Command to display plural value in section title and table of contents in all upper case.

```

12505 \newrobustcmd*{\GLSxtrtitleplural}[1]{%
12506 \GLSplural[noindex,hyper=false]{#1}[]%
12507 }

```

glsxtrheadfirst As above but for the first value.

```

12508 \newcommand*{\glsxtrheadfirst}[1]{%
12509 \protect\NoCaseChange
12510 {%
12511 \glsifattribute{#1}{headuc}{true}%
12512 {%
12513 \GLSfirst[noindex,hyper=false]{#1}[]%
12514 }%

```



```

12515   {%
12516     \glsfirst[noindex,hyper=false]{#1}[]%
12517   }%
12518 }%
12519 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```

12520 \newrobustcmd*{\lsxtrtitlefirst}[1]{%
12521   \glsfirst[noindex,hyper=false]{#1}[]%
12522 }

```

Glsxtrheadfirst First letter converted to upper case

```

12523 \newcommand*{\Glsxtrheadfirst}[1]{%
12524   \protect\NoCaseChange
12525   {%
12526     \glsifattribute{#1}{headuc}{true}%
12527     {%
12528       \GLSfirst[noindex,hyper=false]{#1}[]%
12529     }%
12530     {%
12531       \Glsfirst[noindex,hyper=false]{#1}[]%
12532     }%
12533   }%
12534 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```

12535 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
12536   \Glsfirst[noindex,hyper=false]{#1}[]%
12537 }

```

LSxtrtitlefirst Command to display first value in section title and table of contents in all upper case.

```

12538 \newrobustcmd*{\GLSxtrtitlefirst}[1]{%
12539   \GLSfirst[noindex,hyper=false]{#1}[]%
12540 }

```

headfirstplural As above but for the firstplural value.

```

12541 \newcommand*{\glsxtrheadfirstplural}[1]{%
12542   \protect\NoCaseChange
12543   {%
12544     \glsifattribute{#1}{headuc}{true}%
12545     {%
12546       \GLSfirstplural[noindex,hyper=false]{#1}[]%
12547     }%
12548     {%
12549       \glsfirstplural[noindex,hyper=false]{#1}[]%
12550     }%
12551   }%
12552 }

```

`itlefirstplural` Command to display firstplural value in section title and table of contents.

```
12553 \newrobustcmd*{\glxstrtitlefirstplural}[1]{%
12554   \glsfirstplural[noindex,hyper=false]{#1}[]%
12555 }
```

`headfirstplural` First letter converted to upper case

```
12556 \newcommand*{\Glsxtrheadfirstplural}[1]{%
12557   \protect\NoCaseChange
12558   {%
12559     \glusifattribute{#1}{headuc}{true}%
12560     {%
12561       \GLSfirstplural[noindex,hyper=false]{#1}[]%
12562     }%
12563     {%
12564       \Glsfirstplural[noindex,hyper=false]{#1}[]%
12565     }%
12566   }%
12567 }
```

`itlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
12568 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
12569   \Glsfirstplural[noindex,hyper=false]{#1}[]%
12570 }
```

`itlefirstplural` Command to display first value in section title and table of contents in all upper case.

```
12571 \newrobustcmd*{\GLSxtrtitlefirstplural}[1]{%
12572   \GLSfirstplural[noindex,hyper=false]{#1}[]%
12573 }
```

`\glxstrheadlong` Command used to display long form in the page header.

```
12574 \newcommand*{\glxstrheadlong}[1]{%
12575   \protect\NoCaseChange
12576   {%
12577     \glusifattribute{#1}{headuc}{true}%
12578     {%
12579       \GLSxtrlong[noindex,hyper=false]{#1}[]%
12580     }%
12581     {%
12582       \glsxtrlong[noindex,hyper=false]{#1}[]%
12583     }%
12584   }%
12585 }
```

`glxstrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
12586 \newrobustcmd*{\glxstrtitlelong}[1]{%
12587   \glsxtrlong[noindex,hyper=false]{#1}[]%
12588 }
```

`\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.

```

12589 \newcommand*{\glxtrheadlongpl}[1]{%
12590   \protect\NoCaseChange
12591   {%
12592     \glsifattribute{#1}{headuc}{true}%
12593     {%
12594       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
12595     }%
12596     {%
12597       \glxtrlongpl[noindex,hyper=false]{#1}[]%
12598     }%
12599   }%
12600 }
```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```

12601 \newrobustcmd*{\glxtrtitlelongpl}[1]{%
12602   \glxtrlongpl[noindex,hyper=false]{#1}[]%
12603 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

12604 \newcommand*{\Glsxtrheadlong}[1]{%
12605   \protect\NoCaseChange
12606   {%
12607     \glsifattribute{#1}{headuc}{true}%
12608     {%
12609       \GLSxtrlong[noindex,hyper=false]{#1}[]%
12610     }%
12611     {%
12612       \Glsxtrlong[noindex,hyper=false]{#1}[]%
12613     }%
12614   }%
12615 }
```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12616 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
12617   \Glsxtrlong[noindex,hyper=false]{#1}[]%
12618 }
```

`GLSxtrtitlelong` Command to display long form of abbreviation in section title and table of contents in all upper case.

```

12619 \newrobustcmd*{\GLSxtrtitlelong}[1]{%
12620   \GLSxtrlong[noindex,hyper=false]{#1}[]%
12621 }
```

`\lsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```
12622 \newcommand*{\Glsxtrheadlongpl}[1]{%
12623   \protect\NoCaseChange
12624   {%
12625     \glsifattribute{#1}{headuc}{true}%
12626     {%
12627       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
12628     }%
12629     {%
12630       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
12631     }%
12632   }%
12633 }
```

`\sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
12634 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
12635   \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
12636 }
```

`Sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents in all upper case.

```
12637 \newrobustcmd*{\GLSxtrtitlelongpl}[1]{%
12638   \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
12639 }
```

`\glsxtrheadfull` Command used to display full form in the page header.

```
12640 \newcommand*{\glsxtrheadfull}[1]{%
12641   \protect\NoCaseChange
12642   {%
12643     \glsifattribute{#1}{headuc}{true}%
12644     {%
12645       \GLSxtrfull[noindex,hyper=false]{#1}[]%
12646     }%
12647     {%
12648       \glsxtrfull[noindex,hyper=false]{#1}[]%
12649     }%
12650   }%
12651 }
```

`glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents.

```
12652 \newrobustcmd*{\glsxtrtitlefull}[1]{%
12653   \glsxtrfull[noindex,hyper=false]{#1}[]%
12654 }
```

`\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.

```

12655 \newcommand*{\glxstrheadfullpl}[1]{%
12656   \protect\NoCaseChange
12657   {%
12658     \glusifattribute{#1}{headuc}{true}%
12659     {%
12660       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
12661     }%
12662     {%
12663       \glxstrfullpl[noindex,hyper=false]{#1}[]%
12664     }%
12665   }%
12666 }

```

sxstrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents.

```

12667 \newrobustcmd*{\glxstrtitlefullpl}[1]{%
12668   \glxstrfullpl[noindex,hyper=false]{#1}[]%
12669 }

```

\Glsxtrheadfull Command used to display full form in the page header with the first letter converted to upper case.

```

12670 \newcommand*{\Glsxtrheadfull}[1]{%
12671   \protect\NoCaseChange
12672   {%
12673     \glusifattribute{#1}{headuc}{true}%
12674     {%
12675       \GLSxtrfull[noindex,hyper=false]{#1}[]%
12676     }%
12677     {%
12678       \Glsxtrfull[noindex,hyper=false]{#1}[]%
12679     }%
12680   }%
12681 }

```

Glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12682 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
12683   \Glsxtrfull[noindex,hyper=false]{#1}[]%
12684 }

```

GLSxtrtitlefull Command to display full form of abbreviation in section title and table of contents in all upper case.

```

12685 \newrobustcmd*{\GLSxtrtitlefull}[1]{%
12686   \GLSxtrfull[noindex,hyper=false]{#1}[]%
12687 }

```

lsxtrheadfullpl Command used to display plural full form in the page header with the first letter converted to upper case.

```

12688 \newcommand*{\Glsxtrheadfullpl}[1]{%

```

```

12689 \protect\NoCaseChange
12690 {%
12691   \glsifattribute{#1}{headuc}{true}%
12692   {%
12693     \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
12694   }%
12695   {%
12696     \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
12697   }%
12698 }%
12699 }

```

sxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12700 \newrobustcmd*{\GLSxtrtitlefullpl}[1]{%
12701   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
12702 }

```

Sxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents in all upper case.

```

12703 \newrobustcmd*{\GLSxtrtitlefullpl}[1]{%
12704   \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
12705 }

```

\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.

```

12706 \ifdef\texorpdfstring
12707 {
12708   \newcommand*{\glsfmtshort}[1]{%
12709     \texorpdfstring
12710       {\glsxtrtitleshort{#1}}%
12711       {\glsentryshort{#1}}%
12712   }
12713 }
12714 {
12715   \newcommand*{\glsfmtshort}[1]{%
12716     \glsxtrtitleshort{#1}}
12717 }

```

Similarly for the plural version.

```

\glsfmtshortpl
12718 \ifdef\texorpdfstring
12719 {
12720   \newcommand*{\glsfmtshortpl}[1]{%
12721     \texorpdfstring
12722       {\glsxtrtitleshortpl{#1}}%
12723       {\glsentryshortpl{#1}}%
12724   }

```

```

12725 }
12726 {
12727   \newcommand*{\glsfmtshortpl}[1]{%
12728     \glsxtrtitleshortpl{#1}}
12729 }

```

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).

```

12730 \ifdef\texorpdfstring
12731 {
12732   \newcommand*{\Glsfmtshort}[1]{%
12733     \texorpdfstring
12734       {\Glsxtrtitleshort{#1}}%
12735     {\glsentryshort{#1}}%
12736   }
12737 }
12738 {
12739   \newcommand*{\Glsfmtshort}[1]{%
12740     \Glsxtrtitleshort{#1}}
12741 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

12742 \ifdef\texorpdfstring
12743 {
12744   \newcommand*{\Glsfmtshortpl}[1]{%
12745     \texorpdfstring
12746       {\Glsxtrtitleshortpl{#1}}%
12747     {\glsentryshortpl{#1}}%
12748   }
12749 }
12750 {
12751   \newcommand*{\Glsfmtshortpl}[1]{%
12752     \Glsxtrtitleshortpl{#1}}
12753 }

```

`\glsfmtname` As above but for the name value.

```

12754 \ifdef\texorpdfstring
12755 {
12756   \newcommand*{\glsfmtname}[1]{%
12757     \texorpdfstring
12758       {\glsxtrtitlename{#1}}%
12759     {\glsentryname{#1}}%
12760   }
12761 }
12762 {
12763   \newcommand*{\glsfmtname}[1]{%
12764     \glsxtrtitlename{#1}}
12765 }

```

`\Glsfmtname` First letter converted to upper case.

```
12766 \ifdef\texorpdfstring
12767 {
12768   \newcommand*\Glsfmtname[1]{%
12769     \texorpdfstring
12770     {\Glsxtrtitlename{#1}}%
12771     {\glsentryname{#1}}%
12772   }
12773 }
12774 {
12775   \newcommand*\Glsfmtname[1]{%
12776     \Glsxtrtitlename{#1}}
12777 }
```

`\GLSfmtname` All upper case.

```
12778 \ifdef\texorpdfstring
12779 {
12780   \newcommand*\GLSfmtname[1]{%
12781     \texorpdfstring
12782     {\GLSxtrtitlename{#1}}%
12783     {\glsentryname{#1}}%
12784   }
12785 }
12786 {
12787   \newcommand*\GLSfmtname[1]{%
12788     \GLSxtrtitlename{#1}}
12789 }
```

`\glsfmttext` As above but for the text value.

```
12790 \ifdef\texorpdfstring
12791 {
12792   \newcommand*\glsfmttext[1]{%
12793     \texorpdfstring
12794     {\glsxtrtitletext{#1}}%
12795     {\glsentrytext{#1}}%
12796   }
12797 }
12798 {
12799   \newcommand*\glsfmttext[1]{%
12800     \glsxtrtitletext{#1}}
12801 }
```

`\Glsfmttext` First letter converted to upper case.

```
12802 \ifdef\texorpdfstring
12803 {
12804   \newcommand*\Glsfmttext[1]{%
12805     \texorpdfstring
12806     {\Glsxtrtitletext{#1}}%
12807     {\glsentrytext{#1}}%
```



```

12808 }
12809 }
12810 {
12811   \newcommand*{\Glsfmttext}[1]{%
12812     \Glsxtrtitletext{#1}}
12813 }

```

`\Glsfmttext` All upper case.

```

12814 \ifdef\texorpdfstring
12815 {
12816   \newcommand*{\Glsfmttext}[1]{%
12817     \texorpdfstring
12818       {\Glsxtrtitletext{#1}}%
12819       {\glsentrytext{#1}}%
12820   }
12821 }
12822 {
12823   \newcommand*{\Glsfmttext}[1]{%
12824     \Glsxtrtitletext{#1}}
12825 }

```

`\glsfmtplural` As above but for the plural value.

```

12826 \ifdef\texorpdfstring
12827 {
12828   \newcommand*{\glsfmtplural}[1]{%
12829     \texorpdfstring
12830       {\glsxtrtitleplural{#1}}%
12831       {\glsentryplural{#1}}%
12832   }
12833 }
12834 {
12835   \newcommand*{\glsfmtplural}[1]{%
12836     \glsxtrtitleplural{#1}}
12837 }

```

`\Glsfmtplural` First letter converted to upper case.

```

12838 \ifdef\texorpdfstring
12839 {
12840   \newcommand*{\Glsfmtplural}[1]{%
12841     \texorpdfstring
12842       {\Glsxtrtitleplural{#1}}%
12843       {\glsentryplural{#1}}%
12844   }
12845 }
12846 {
12847   \newcommand*{\Glsfmtplural}[1]{%
12848     \Glsxtrtitleplural{#1}}
12849 }

```

`\GLSfmtplural` All upper case.

```
12850 \ifdef\txorpdfstring
12851 {
12852   \newcommand*{\GLSfmtplural}[1]{%
12853     \txorpdfstring
12854     {\GLSxtrtitleplural{#1}}%
12855     {\glsentryplural{#1}}%
12856   }
12857 }
12858 {
12859   \newcommand*{\GLSfmtplural}[1]{%
12860     \GLSxtrtitleplural{#1}}
12861 }
```

`\glsfmtfirst` As above but for the first value.

```
12862 \ifdef\txorpdfstring
12863 {
12864   \newcommand*{\glsfmtfirst}[1]{%
12865     \txorpdfstring
12866     {\glsxtrtitlefirst{#1}}%
12867     {\glsentryfirst{#1}}%
12868   }
12869 }
12870 {
12871   \newcommand*{\glsfmtfirst}[1]{%
12872     \glsxtrtitlefirst{#1}}
12873 }
```

`\Glsfmtfirst` First letter converted to upper case.

```
12874 \ifdef\txorpdfstring
12875 {
12876   \newcommand*{\Glsfmtfirst}[1]{%
12877     \txorpdfstring
12878     {\Glsxtrtitlefirst{#1}}%
12879     {\glsentryfirst{#1}}%
12880   }
12881 }
12882 {
12883   \newcommand*{\Glsfmtfirst}[1]{%
12884     \Glsxtrtitlefirst{#1}}
12885 }
```

`\GLSfmtfirst` All upper case.

```
12886 \ifdef\txorpdfstring
12887 {
12888   \newcommand*{\GLSfmtfirst}[1]{%
12889     \txorpdfstring
12890     {\GLSxtrtitlefirst{#1}}%
12891     {\glsentryfirst{#1}}%
```

```

12892 }
12893 }
12894 {
12895   \newcommand*{\GLSfmtfirst}[1]{%
12896     \Glsxtrtitlefirst{#1}}
12897 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

12898 \ifdef\texorpdfstring
12899 {
12900   \newcommand*{\glsfmtfirstpl}[1]{%
12901     \texorpdfstring
12902       {\Glsxtrtitlefirstplural{#1}}%
12903       {\glsentryfirstplural{#1}}%
12904   }
12905 }
12906 {
12907   \newcommand*{\glsfmtfirstpl}[1]{%
12908     \Glsxtrtitlefirstplural{#1}}
12909 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

12910 \ifdef\texorpdfstring
12911 {
12912   \newcommand*{\Glsfmtfirstpl}[1]{%
12913     \texorpdfstring
12914       {\Glsxtrtitlefirstplural{#1}}%
12915       {\glsentryfirstplural{#1}}%
12916   }
12917 }
12918 {
12919   \newcommand*{\Glsfmtfirstpl}[1]{%
12920     \Glsxtrtitlefirstplural{#1}}
12921 }

```

`\GLSfmtfirstpl` All upper case.

```

12922 \ifdef\texorpdfstring
12923 {
12924   \newcommand*{\GLSfmtfirstpl}[1]{%
12925     \texorpdfstring
12926       {\GLSxtrtitlefirstplural{#1}}%
12927       {\glsentryfirstplural{#1}}%
12928   }
12929 }
12930 {
12931   \newcommand*{\GLSfmtfirstpl}[1]{%
12932     \GLSxtrtitlefirstplural{#1}}
12933 }

```

`\glsfmtlong` As above but for the long value.

```
12934 \ifdef\txorpdfstring
12935 {
12936   \newcommand*\glsfmtlong[1]{%
12937     \txorpdfstring
12938     {\glsxtrtitlelong{#1}}%
12939     {\glsentrylong{#1}}%
12940   }
12941 }
12942 {
12943   \newcommand*\glsfmtlong[1]{%
12944     \glsxtrtitlelong{#1}}
12945 }
```

`\Glsfmtlong` First letter converted to upper case.

```
12946 \ifdef\txorpdfstring
12947 {
12948   \newcommand*\Glsfmtlong[1]{%
12949     \txorpdfstring
12950     {\Glsxtrtitlelong{#1}}%
12951     {\glsentrylong{#1}}%
12952   }
12953 }
12954 {
12955   \newcommand*\Glsfmtlong[1]{%
12956     \Glsxtrtitlelong{#1}}
12957 }
```

`\GLSfmtlong` All upper case.

```
12958 \ifdef\txorpdfstring
12959 {
12960   \newcommand*\GLSfmtlong[1]{%
12961     \txorpdfstring
12962     {\GLSxtrtitlelong{#1}}%
12963     {\glsentrylong{#1}}%
12964   }
12965 }
12966 {
12967   \newcommand*\GLSfmtlong[1]{%
12968     \GLSxtrtitlelong{#1}}
12969 }
```

`\glsfmtlongpl` As above but for the longplural value.

```
12970 \ifdef\txorpdfstring
12971 {
12972   \newcommand*\glsfmtlongpl[1]{%
12973     \txorpdfstring
12974     {\glsxtrtitlelongpl{#1}}%
12975     {\glsentrylongpl{#1}}%
12976   }
12977 }
```

```

12976 }
12977 }
12978 {
12979   \newcommand*{\glsfmtlongpl}[1]{%
12980     \glsxtrtitlelongpl{#1}}
12981 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

12982 \ifdef\teorpdfstring
12983 {
12984   \newcommand*{\Glsfmtlongpl}[1]{%
12985     \teorpdfstring
12986       {\Glsxtrtitlelongpl{#1}}%
12987       {\glsentrylongpl{#1}}%
12988   }
12989 }
12990 {
12991   \newcommand*{\Glsfmtlongpl}[1]{%
12992     \Glsxtrtitlelongpl{#1}}
12993 }

```

`\GLSfmtlongpl` All upper case.

```

12994 \ifdef\teorpdfstring
12995 {
12996   \newcommand*{\GLSfmtlongpl}[1]{%
12997     \teorpdfstring
12998       {\GLSxtrtitlelongpl{#1}}%
12999       {\glsentrylongpl{#1}}%
13000   }
13001 }
13002 {
13003   \newcommand*{\GLSfmtlongpl}[1]{%
13004     \GLSxtrtitlelongpl{#1}}
13005 }

```

`\glspdffmtfull` Can't use `\glsxtrinlinefullformat` in PDF bookmarks as it's not fully expandable. This command is for the PDF part of `\teorpdfstring` for the full form.

```

13006 \newcommand*{\glspdffmtfull}[1]{\glsentrylong{#1} (\glsentryshort{#1})}%

```

`glspdffmtfullpl` Likewise for plural.

```

13007 \newcommand*{\glspdffmtfullpl}[1]{\glsentrylongpl{#1} (\glsentryshortpl{#1})}%

```

`\glsfmtfull` In-line full format.

```

13008 \ifdef\teorpdfstring
13009 {
13010   \newcommand*{\glsfmtfull}[1]{%
13011     \teorpdfstring
13012       {\glsxtrtitlefull{#1}}%

```

```

13013     {\glspdffmtfull{#1}}%
13014 }
13015 }
13016 {
13017   \newcommand*{\glsfmtfull}[1]{%
13018     \glstrtitlefull{#1}}
13019 }

```

`\Glsfmtfull` First letter converted to upper case.

```

13020 \ifdef\teorpdfstring
13021 {
13022   \newcommand*{\Glsfmtfull}[1]{%
13023     \teorpdfstring
13024     {\Glsxtrtitlefull{#1}}%
13025     {\glspdffmtfull{#1}}}%
13026 }
13027 }
13028 {
13029   \newcommand*{\Glsfmtfull}[1]{%
13030     \Glsxtrtitlefull{#1}}
13031 }

```

`\GLSfmtfull` All upper case.

```

13032 \ifdef\teorpdfstring
13033 {
13034   \newcommand*{\GLSfmtfull}[1]{%
13035     \teorpdfstring
13036     {\GLSxtrtitlefull{#1}}%
13037     {\glspdffmtfull{#1}}}%
13038 }
13039 }
13040 {
13041   \newcommand*{\GLSfmtfull}[1]{%
13042     \GLSxtrtitlefull{#1}}
13043 }

```

`\glsfmtfullpl` In-line full plural format.

```

13044 \ifdef\teorpdfstring
13045 {
13046   \newcommand*{\glsfmtfullpl}[1]{%
13047     \teorpdfstring
13048     {\glsxtrtitlefullpl{#1}}%
13049     {\glspdffmtfullpl{#1}}}%
13050 }
13051 }
13052 {
13053   \newcommand*{\glsfmtfullpl}[1]{%
13054     \glsxtrtitlefullpl{#1}}
13055 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```
13056 \ifdef\txorpdfstring
13057 {
13058   \newcommand*\Glsfmtfullpl}[1]{%
13059     \txorpdfstring
13060     {\Glsxtrtitlefullpl{#1}}%
13061     {\glspdfdfmtfullpl{#1}{}}%
13062   }
13063 }
13064 {
13065   \newcommand*\Glsfmtfullpl}[1]{%
13066     \Glsxtrtitlefullpl{#1}}
13067 }
```

`\GLSfmtfullpl` All upper case.

```
13068 \ifdef\txorpdfstring
13069 {
13070   \newcommand*\GLSfmtfullpl}[1]{%
13071     \txorpdfstring
13072     {\GLSxtrtitlefullpl{#1}}%
13073     {\glspdfdfmtfullpl{#1}{}}%
13074   }
13075 }
13076 {
13077   \newcommand*\GLSfmtfullpl}[1]{%
13078     \GLSxtrtitlefullpl{#1}}
13079 }
```

1.9 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```
13080 \newcommand*\RequireGlossariesExtraLang}[1]{%
13081   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
13082 }
```

`sariesExtraLang`

```
13083 \newcommand*\ProvidesGlossariesExtraLang}[1]{%
13084   \ProvidesFile{glossariesxtr-#1.ldf}%
13085 }
```

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.

```
13086 \newcommand{\glxtr@loaddialect}{%
13087   \IfTrackedLanguageFileExists{\this@dialect}%
13088   {\glossariesxtr-}% prefix
13089   {\ldf}%
13090   {%
13091     \RequireGlossariesExtraLang{\CurrentTrackedTag}%
13092   }%
13093   {}% not found
```

If `glossaries-extra-bib2gls` has been loaded, `\@glxtrdialecthook` will check for the associated script, otherwise it will do nothing.

```
13094 \@glxtrdialecthook
13095 }
```

```
13096 \@ifpackageloaded{tracklang}
13097 {%
13098   \AnyTrackedLanguages
13099   {%
13100     \ForEachTrackedDialect{\this@dialect}{\glxtr@loaddialect}%
13101   }%
13102   {}%
13103 }
13104 {}
```

Load `glossaries-extra-stylemods` if required.

```
13105 \@glxtr@redefstyles
```

and set the style:

```
13106 \@glxtr@do@style
```

1.10 glossaries-extra-bib2gls.sty

This package provides additional support for `bib2gls` and is automatically loaded by the `record` option.

```
13107 \NeedsTeXFormat{LaTeX2e}
13108 \ProvidesPackage{glossaries-extra-bib2gls}[2021/09/20 v1.46 (NLCT)]
```

Provide convenient shortcut commands for predefined glossary types.

`printunsrtacronyms`

```
13109 \ifglssacronym
13110   \providecommand*\printunsrtacronyms[1][1]{%
13111     \printunsrtglossary[type=\acronymtype,#1]}%
13112 \fi
```

`printunsrtindex`

```
13113 \ifglossaryexists{index}
13114 {
```



```

13115 \providecommand*\printunsrtindex}[1][]{%
13116 \printunsrtglossary[type=index,#1]}%
13117 }{}

```

intunsrtsymbols

```

13118 \ifglossaryexists{symbols}
13119 {
13120 \providecommand*\printunsrtsymbols}[1][]{%
13121 \printunsrtglossary[type=symbols,#1]}%
13122 }{}

```

intunsrtnumbers

```

13123 \ifglossaryexists{numbers}
13124 {
13125 \providecommand*\printunsrtnumbers}[1][]{%
13126 \printunsrtglossary[type=numbers,#1]}%
13127 }{}

```

rtaabbreviations

```

13128 \ifglossaryexists{abbreviations}
13129 {
13130 \providecommand*\printunsrtaabbreviations}[1][]{%
13131 \printunsrtglossary[type=abbreviations,#1]}%
13132 }{}

```

splaynumberlist Allow \glsdisplaynumberlist and make it robust.

```

13133 \renewcommand*\glsdisplaynumberlist}[1]{%
13134 \glsdoifexists{#1}%
13135 {%
13136 {\let\bibglsgldelimN\glsnumlistsep
13137 \let\bibglsglastDelimN\glsnumlistlastsep
13138 \glsxtrusefield{#1}{location}%
13139 }%
13140 }%
13141 }
13142 \robustify\glsdisplaynumberlist

```

entrynumberlist

```

13143 \renewcommand*\glsentrynumberlist}[1]{\glsxtrusefield{#1}{location}}

```

These are some convenient macros for use with custom rules.

\glshex

```

13144 \newcommand*\glshex{\string\u}

```

lscapturedgroup

```

13145 \newcommand*\glsapturedgroup{\string\$}

```

nZeroChildCount For use with bib2gls's save-child-count resource option.

```
13146 \newcommand*{\GlsXtrIfHasNonZeroChildCount}[3]{%
13147   \GlsXtrIfFieldNonZero{childcount}{#1}{#2}{#3}%
13148 }
```

rprovidecommand For use in @preamble, this behaves like \providecommand in the document but like \renewcommand in bib2gls.

```
13149 \newcommand*{\glstrprovidecommand}{\providecommand}
```

glsrenewcommand Like \renewcommand but only generates a warning rather than an error if the command isn't defined.

```
13150 \newcommand*{\glstrenewcommand}{\@star@or@long\glstr@renewcommand}
```

tr@renewcommand

```
13151 \newcommand*{\glstr@renewcommand}[1]{%
13152   \begingroup \escapechar\m@ne\xdef\@gtempa{\string#1}\endgroup
13153   \expandafter\@ifundefined\@gtempa
13154     {%
13155       \GlossariesExtraWarning{can't redefine \noexpand#1(not already defined)}%
13156     }%
13157   \relax
13158   \relax
13159   \let\@ifdefinable\@rc@ifdefinable
13160   \new@command#1%
13161 }
```

lossarylocation For use with indexcounter and bib2gls.

```
13162 \newcommand*{\glstr@wrglossarylocation}[2]{#1}
```

indexCounterLink

`\GlsXtrIndexCounterLink{<text>}{<label>}`

For use with indexcounter and bib2gls.

```
13163 \ifdef\hyperref
13164 {%
13165   \newcommand*{\GlsXtrIndexCounterLink}[2]{%
13166     \glstrifhasfield{indexcounter}{#2}%
13167     {\hyperref[wrglossary.\glscurrentfieldvalue]{#1}}%
13168     {#1}%
13169   }
13170 }
13171 {
13172   \newcommand*{\GlsXtrIndexCounterLink}[2]{#1}
13173 }
```

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.

```
13174 \newcommand*{\GlsXtrDualField}{dual}
```

XtrDualBackLink

`\GlsXtrDualBackLink{<text>}{<label>}`

Adds a hyperlink to the dual entry.

```
13175 \newcommand*{\GlsXtrDualBackLink}[2]{%
13176   \glstrifhasfield{\GlsXtrDualField}{#2}%
13177   {\glshyperlink[#1]{\glscurrentfieldvalue}}%
13178   {#2}%
13179 }
```

TeXEntryAliases Convenient shortcut for use with entry-type-aliases to alias standard BibTeX entry types to @bibtexentry.

```
13180 \newcommand*{\GlsXtrBibTeXEntryAliases}{%
13181   article=bibtexentry,
13182   book=bibtexentry,
13183   booklet=bibtexentry,
13184   conference=bibtexentry,
13185   inbook=bibtexentry,
13186   incollection=bibtexentry,
13187   inproceedings=bibtexentry,
13188   manual=bibtexentry,
13189   mastersthesis=bibtexentry,
13190   misc=bibtexentry,
13191   phdthesis=bibtexentry,
13192   proceedings=bibtexentry,
13193   techreport=bibtexentry,
13194   unpublished=bibtexentry
13195 }
```

ideBibTeXFields Convenient shortcut to define the standard BibTeX fields.

```
13196 \newcommand*{\GlsXtrProvideBibTeXFields}{%
13197   \glsaddstoragekey{address}{\glstrbibaddress}%
13198   \glsaddstoragekey{author}{\glstrbibauthor}%
13199   \glsaddstoragekey{booktitle}{\glstrbibbooktitle}%
13200   \glsaddstoragekey{chapter}{\glstrbibchapter}%
13201   \glsaddstoragekey{edition}{\glstrbibedition}%
13202   \glsaddstoragekey{howpublished}{\glstrbibhowpublished}%
13203   \glsaddstoragekey{institution}{\glstrbibinstitution}%

```

```

13204 \glsaddstoragekey{journal}{\glsxtrbibjournal}%
13205 \glsaddstoragekey{month}{\glsxtrbibmonth}%
13206 \glsaddstoragekey{note}{\glsxtrbibnote}%
13207 \glsaddstoragekey{number}{\glsxtrbibnumber}%
13208 \glsaddstoragekey{organization}{\glsxtrbiborganization}%
13209 \glsaddstoragekey{pages}{\glsxtrbibpages}%
13210 \glsaddstoragekey{publisher}{\glsxtrbibpublisher}%
13211 \glsaddstoragekey{school}{\glsxtrbibschooll}%
13212 \glsaddstoragekey{series}{\glsxtrbibseries}%
13213 \glsaddstoragekey{title}{\glsxtrbibtitle}%
13214 \glsaddstoragekey{bibtextype}{\glsxtrbibtype}%
13215 \glsaddstoragekey{volume}{\glsxtrbibvolume}%
13216 }

```

Multiple supplementary references are only supported with bib2gls.

`\glstisuppllocation` 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.

```

13217 \newcommand*\glsxtrmultisuppllocation[3]{%
13218   {%
13219     \def\glsxtrsuppllocationurl{#2}%
13220     \glshypernumber{#1}%
13221   }%
13222 }

```

`\glstrdisplaysupplloc`

`\glstrdisplaysupplloc{<prefix>}{<counter>}{<format>}{<src>}{<location>}`

This is like `\glsnoidxdisplayloc` but is used for supplementary locations and so requires an extra argument.

```

13223 \newcommand*\glstrdisplaysupplloc[5]{%
13224   \setentrycounter[#1]{#2}%
13225   \glsxtrmultisuppllocation{#5}{#4}{#3}%
13226 }

```

`\glstrdisplaylocnameref` `\glstrdisplaylocnameref{<prefix>}{<counter>}{<format>}{<location>}{<name>}{<href>}{<hcounter>}{<external file>}` Used with the `[nameref]` record package option. The `<href>` argument was obtained from `\@currentHref` and the `<hcounter>` argument was obtained from `\theHentrycounter`, which is more reliable. If `hyperref` hasn't been loaded, this just behaves like `\glsnoidxdisplayloc`.

```

13227 \ifundef\hyperlink
13228 {
13229   \newcommand*\glstrdisplaylocnameref[8]{%
13230     \glsnoidxdisplayloc{#1}{#2}{#3}{#4}%

```

```

13231 }
13232 }
13233 {

```

Default action uses $\langle hcounter \rangle$. 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 $\backslash\mathrm{glstr}\langle counter \rangle\mathrm{locfmt}$ is defined.

```

13234 \newcommand*\glstrdisplaylocnameref}[8]{%
13235   \ifcsdef{glstr#2locfmt}%
13236     {\glstrnameref link{#3}{\csuse{glstr#2locfmt}{#4}{#5}}{#2.#7}{#8}}%
13237     {%
13238       \ifstrempy{#5}%
13239       {%

```

No title, so just use the location as the link text.

```

13240   \glstrnameref link{#3}{#4}{#2.#7}{#8}}%
13241   }%
13242   {%
13243     \ifstrequal{#2}{page}%
13244     {\glstrnameref link{#3}{#4}{#2.#7}{#8}}%
13245     {\glstrnameref link{#3}{#5}{#2.#7}{#8}}%
13246   }%
13247 }%
13248 }
13249 }

```

re equationlocfmt

```
\glstrequationlocfmt{\langle location \rangle}{\langle title \rangle}
```

```

13250 \newcommand*\glstrequationlocfmt}[2]{(\#1)}

```

sxtrnameref link

```
\glstrfmt nameref link{\langle format \rangle}{\langle title \rangle}{\langle href \rangle}{\langle external file \rangle}
```

```

13251 \newcommand*\glstrnameref link}[4]{%

```

Locally change $\backslash\mathrm{glshypernumber}$ to $\backslash\mathrm{@firstofone}$ to remove the normal location hyperlink.

```

13252 \begin group
13253 \let\glshypernumber\@firstofone

```

If the $\langle external file \rangle$ argument is empty, an internal link is used, otherwise an external one is needed.

```

13254 \ifstrempy{#4}%
13255 {\glstrfmt internal nameref{#3}{#1}{#2}}%

```

```

13256     {\glxstrfmtexternalnameref{#3}{#1}{#2}{#4}}%
13257 \endgroup
13258 }

```

sxtrnameloclink

```

\glxstrnamerefloclink{<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.

```

13259 \newcommand{\glxstrnameloclink}[6]{%
13260 \begingroup
13261 \setentrycounter[#1]{#2}%
13262 \def\glxstr@locationhypertext{#5}%
13263 \let\glshypernumber\@firstofone
13264 \def\@glsnumberformat{#3}%
13265 \def\glxstrsupplocationurl{#6}%
13266 \toks@={}%
13267 \@glxstr@bibgls@removespaces#4 \@nil
13268 \endgroup
13269 }

```

ls@removespaces

```

13270 \def\@glxstr@bibgls@removespaces#1 #2\@nil{%
13271 \toks@=\expandafter{\the\toks@#1}%
13272 \ifx\#2\%
13273 \edef\@glo@tmp{\the\toks@}%
13274 \ifx\@glo@tmp\empty
13275 \else
13276 \protected@edef\@glo@tmp{\glstentrycounter\@glo@counterprefix\the\toks@}%
13277 \ifdefvoid\glxstrsupplocationurl
13278 {%
13279 \expandafter\glxstrfmtinternalnameref\expandafter{\@glo@tmp}%
13280 {\@glsnumberformat}{\glxstr@locationhypertext}%
13281 }%
13282 {%
13283 \expandafter\glxstrfmtexternalnameref\expandafter{\@glo@tmp}%
13284 {\@glsnumberformat}{\glxstr@locationhypertext}{\glxstrsupplocationurl}%
13285 }%
13286 \fi
13287 \else
13288 \@gls@ReturnAfterFi{%
13289 \@glxstr@bibgls@removespaces#2\@nil
13290 }%
13291 \fi

```

13292 }

internalnameref

```
\glstrfmtinternalnameloc{<target>}{<format>}{<title>}
```

```
13293 \newcommand*{\glstrfmtinternalnameref}[3]{%
13294   \csuse{#2}{\glsdohyperlink{#1}{#3}}%
13295 }
```

externalnameref

```
\glstrfmtexternalnameloc{<target>}{<format>}{<title>}{<file>}
```

```
13296 \newcommand*{\glstrfmtexternalnameref}[4]{%
13297   \csuse{#2}{\hyperref{#4}{#1}{#3}}%
13298 }
```

glstrSetWidest

```
\glstrSetWidest{<type>}{<level>}{<text>}
```

As from **bib2gls** v1.8, this is used by the set-widest resource option for the altree and the styles provided by the glossary-longextra package.

```
13299 \newcommand*{\glstrSetWidest}[3]{%
```

Check which style options have been provided. (The style packages may not have been loaded.)

```
13300   \ifdef\glupdatewidest
13301   {%
13302     \ifdef\glslongextraUpdateWidest
13303     {%
```

Relevant style packages all loaded. If the *<type>* has been given, append to glossary preamble.

```
13304     \ifstrempy{#1}
13305     {%
13306       \glupdatewidest[#2]{#3}%
13307       \ifnum#2=0\relax
13308       \glslongextraUpdateWidest{#3}%
13309     \else
13310       \glslongextraUpdateWidestChild{#2}{#3}%
13311     \fi
13312   }%
13313   {%
13314     \apptoglossarypreamble[#1]{\glupdatewidest[#2]{#3}}%
```

```

13315      \ifnum#2=0\relax
13316      \apptoglossarypreamble[#1]{\glslongextraUpdateWidest{#3}}%
13317      \else
13318      \apptoglossarypreamble[#1]{\glslongextraUpdateWidestChild{#2}{#3}}%
13319      \fi
13320    }%
13321  }%
13322  {%

```

Only altree.

```

13323      \ifstrempy{#1}
13324      {%
13325      \glsupdatewidest[#2]{#3}%
13326      }%
13327      {%
13328      \apptoglossarypreamble[#1]{\glsupdatewidest[#2]{#3}}%
13329      }%
13330    }%
13331  }%
13332  {%

```

\glsupdatewidest hasn't been defined. This could just mean that the glossaries-extra-stylemods package hasn't been loaded.

```

13333      \ifdef\glssetwidest
13334      {%
13335      \ifdef\glslongextraUpdateWidest
13336      {%

```

Relevant glossary-tree and glossary-longextra have been loaded. If the *<type>* has been given, append to glossary preamble.

```

13337      \ifstrempy{#1}
13338      {%
13339      \glssetwidest[#2]{#3}%
13340      \ifnum#2=0\relax
13341      \glslongextraUpdateWidest{#3}%
13342      \else
13343      \glslongextraUpdateWidestChild{#2}{#3}%
13344      \fi
13345      }%
13346      {%
13347      \apptoglossarypreamble[#1]{\glssetwidest[#2]{#3}}%
13348      \ifnum#2=0\relax
13349      \apptoglossarypreamble[#1]{\glslongextraUpdateWidest{#3}}%
13350      \else
13351      \apptoglossarypreamble[#1]{\glslongextraUpdateWidestChild{#2}{#3}}%
13352      \fi
13353      }%
13354      }%
13355      {%

```

Only altree.


```

13356     \ifstrempy{#1}
13357     {%
13358         \glsetwidest[#2]{#3}%
13359     }%
13360     {%
13361         \apptoglossarypreamble[#1]{\glsetwidest[#2]{#3}}%
13362     }%
13363 }%
13364 }%
13365 {%
13366     \ifdef\glslongextraUpdateWidest
13367     {%
        glossary-longextra has been loaded.
13368     \ifstrempy{#1}
13369     {%
13370         \ifnum#2=0\relax
13371             \glslongextraUpdateWidest{#3}%
13372         \else
13373             \glslongextraUpdateWidestChild{#2}{#3}%
13374         \fi
13375     }%
13376     {%
13377         \ifnum#2=0\relax
13378             \apptoglossarypreamble[#1]{\glslongextraUpdateWidest{#3}}%
13379         \else
13380             \apptoglossarypreamble[#1]{\glslongextraUpdateWidestChild{#2}{#3}}%
13381         \fi
13382     }%
13383 }%
        Neither glossary-tree nor glossary-longextra have been loaded. Do nothing.
13384     }%
13385 }%
13386 }%
13387 }

```

tWidestFallback

```
\glxtrSetWidestFallback{<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).

```

13388 \newcommand*{\glxtrSetWidestFallback}[2]{%
13389     \ifnum#1=0\relax
13390     \ifdef\glsFindWidestTopLevelName
13391     {%

```

```

13392     \glsFindWidestTopLevelName[#2]%
13393 }%
13394 {%
13395     \GlossariesExtraWarning{You need stylemods={tree} to
13396         provide a fallback for set-widest}%
13397 }%
13398 \else
13399     \ifdef\glsFindWidestLevelTwo
13400     {%
13401         \glsFindWidestLevelTwo[#2]%
13402         \ifdef\glslongextraUpdateWidestChild
13403         {%
13404             \glslongextraUpdateWidestChild{#1}{\csuse{@glswidestnamei}}%
13405             \glslongextraUpdateWidestChild{#1}{\csuse{@glswidestnameii}}%
13406         }%
13407     }%
13408 }%
13409 {%
13410     \GlossariesExtraWarning{You need stylemods={tree} to
13411         provide a fallback for set-widest}%
13412 }%
13413 \fi
13414 }

```

`r@labelprefixes` List of label prefixes.

```

13415 \newcommand*{\@glxstr@labelprefixes}{}

```

`arlabelprefixes` List of label prefixes.

```

13416 \newcommand*{\glxstrclearlabelprefixes}{%
13417     \renewcommand*{\@glxstr@labelprefixes}{}%
13418 }

```

`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.

```

13419 \newcommand*{\glxstraddlabelprefix}[1]{%
13420     \ifstrempy{#1}%
13421     {\glxstraddlabelprefix{\empty}}%
13422     {%
13423         \ifdefempty\@glxstr@labelprefixes
13424         {\def\@glxstr@labelprefixes{#1}}%
13425         {\appto\@glxstr@labelprefixes{,#1}}%
13426     }%
13427 }

```

`pendlabelprefix` Inserts at the start of the list.

```

13428 \newcommand*{\glxstrprependlabelprefix}[1]{%
13429     \ifstrempy{#1}%

```

```

13430 {\glxstrprependlabelprefix{\empty}}%
13431 {%
13432   \ifdefempty\@glxstr@labelprefixes
13433   {\def\@glxstr@labelprefixes{#1}}%
13434   {\preto\@glxstr@labelprefixes{#1,}}%
13435 }%
13436 }

```

labelprefixlist

`\glxstrifinlabelprefixlist{<prefix>}{<true>}{<false>}`

Test if the given prefix is in the list.

```

13437 \newcommand*{\glxstrifinlabelprefixlist}[3]{%
13438   \ifstrempy{#1}%
13439   {\glxstrifinlabelprefixlist{\empty}{#2}{#3}}%
13440   {%
13441     \DTLifinlist{#1}{\@glxstr@labelprefixes}{#2}{#3}%
13442   }%
13443 }

```

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**.

```

13444 \AtBeginDocument{%
13445   \protected@write\@auxout{}{\string\providecommand{\string\@glxstr@prefixlabellist}[1]{}}%
13446   \protected@write\@auxout{}{\string\@glxstr@prefixlabellist{\@glxstr@labelprefixes}}%
13447 }

```

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 \LaTeX run, so the last prefix in the list needs to be the fallback one. Grouping is used in case of a nested for loop.

```

13448 \newcommand*{\@glxstr@get@prefixedlabel}[1]{%
13449   \begin{group}
13450     \protected@edef\@gls@thislabel{#1}%
13451     \@for\@glxstr@prefix:=\@glxstr@labelprefixes\do
13452     {%
13453       \protected@edef\@gls@thislabel{\@glxstr@prefix#1}%
13454       \ifglstryexists{\@gls@thislabel}{\@endfortrue}{}%
13455     }%
13456     \edef\@glo@tmp{\endgroup\noexpand\def\noexpand\@gls@thislabel{\@gls@thislabel}}\@glo@tmp
13457 }

```

\dgl Like \gls but tries the prefixes. (Can't use \pgl as that's provided by glossaries-prefix.) Since this command is designed for bib2gls's dual entry system, the "d" stands for "dual".

```

13458 \newrobustcmd*{\dgl}{\@gls@hyp@opt\@dgl}

```

```

\@dglS
13459 \newcommand*{\@dglS}[2][\%
13460   \@glSxtr@get@prefixedlabel{#2}%
13461   \new@ifnextchar[{\@glS@{#1}{\@glS@thislabel}}{\@glS@{#1}{\@glS@thislabel}}[]}%
13462 }

\dglSpl
13463 \newrobustcmd*{\dglSpl}{\@glS@hyp@opt\dglSpl}

\@dglSpl
13464 \newcommand*{\@dglSpl}[2][\%
13465   \@glSxtr@get@prefixedlabel{#2}%
13466   \new@ifnextchar[{\@glSpl@{#1}{\@glS@thislabel}}{\@glSpl@{#1}{\@glS@thislabel}}[]}%
13467 }

\dGls
13468 \newrobustcmd*{\dGls}{\@glS@hyp@opt\dGls}

\@dGls
13469 \newcommand*{\@dGls}[2][\%
13470   \@glSxtr@get@prefixedlabel{#2}%
13471   \new@ifnextchar[{\@Gls@{#1}{\@glS@thislabel}}{\@Gls@{#1}{\@glS@thislabel}}[]}%
13472 }

\dGlsPl
13473 \newrobustcmd*{\dGlsPl}{\@glS@hyp@opt\dGlsPl}

\@dGlsPl
13474 \newcommand*{\@dGlsPl}[2][\%
13475   \@glSxtr@get@prefixedlabel{#2}%
13476   \new@ifnextchar[{\@GlsPl@{#1}{\@glS@thislabel}}{\@GlsPl@{#1}{\@glS@thislabel}}[]}%
13477 }

\dGLS
13478 \newrobustcmd*{\dGLS}{\@glS@hyp@opt\dGLS}

\@dGLS
13479 \newcommand*{\@dGLS}[2][\%
13480   \@glSxtr@get@prefixedlabel{#2}%
13481   \new@ifnextchar[{\@GLS@{#1}{\@glS@thislabel}}{\@GLS@{#1}{\@glS@thislabel}}[]}%
13482 }

\dGLSpl
13483 \newrobustcmd*{\dGLSpl}{\@glS@hyp@opt\dGLSpl}

```

`\@dGLSp1`

```
13484 \newcommand*{\@dGLSp1}[2] [] {%
13485   \@glsxtr@get@prefixedlabel{#2}%
13486   \new@ifnextchar[{\@GLSp1@{#1}{\@gls@thislabel}}{\@GLSp1@{#1}{\@gls@thislabel}}[]}%
13487 }
```

`\dglslink` Like `\glslink` but tries the prefixes.

```
13488 \newrobustcmd*{\dglslink}[3] [] {%
13489   \@glsxtr@get@prefixedlabel{#2}%
13490   \glslink[#1]{\@gls@thislabel}{#3}%
13491 }
```

`\dglsdisp` Like `\glsdisp` but tries the prefixes.

```
13492 \newrobustcmd*{\dglsdisp}[3] [] {%
13493   \@glsxtr@get@prefixedlabel{#2}%
13494   \glsdisp[#1]{\@gls@thislabel}{#3}%
13495 }
```

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 \TeX 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`

```
13496 \providecommand*{\Alpha}{\mathrm{A}}
```

`\Beta`

```
13497 \providecommand*{\Beta}{\mathrm{B}}
```

`\Epsilon`

```
13498 \providecommand*{\Epsilon}{\mathrm{E}}
```

`\Zeta`

```
13499 \providecommand*{\Zeta}{\mathrm{Z}}
```

`\Eta`

```
13500 \providecommand*{\Eta}{\mathrm{H}}
```

`\Iota`

```
13501 \providecommand*{\Iota}{\mathrm{I}}
```

`\Kappa`

```
13502 \providecommand*{\Kappa}{\mathrm{K}}
```

`\Mu`

```
13503 \providecommand*{\Mu}{\mathrm{M}}
```

\backslash Nu

13504 \providecommand*{\Nu}{\mathrm{N}}

\backslash Omicron

13505 \providecommand*{\Omicron}{\mathrm{O}}

\backslash Rho

13506 \providecommand*{\Rho}{\mathrm{P}}

\backslash Tau

13507 \providecommand*{\Tau}{\mathrm{T}}

\backslash Chi

13508 \providecommand*{\Chi}{\mathrm{X}}

\backslash Digamma

13509 \providecommand*{\Digamma}{\mathrm{F}}

\backslash omicron

13510 \providecommand*{\omicron}{\mathit{o}}

Provide corresponding upright characters if upgreek has been loaded. (The upper case characters are the same as above.)

13511 \@ifpackageloaded{upgreek}%

13512 {

\backslash Upalpha

13513 \providecommand*{\Upalpha}{\mathrm{A}}

\backslash Upbeta

13514 \providecommand*{\Upbeta}{\mathrm{B}}

\backslash Upsilon

13515 \providecommand*{\Upsilon}{\mathrm{E}}

\backslash Upzeta

13516 \providecommand*{\Upzeta}{\mathrm{Z}}

\backslash Upeta

13517 \providecommand*{\Upeta}{\mathrm{H}}

\backslash Upiota

13518 \providecommand*{\Upiota}{\mathrm{I}}

\backslash Upkappa

13519 \providecommand*{\Upkappa}{\mathrm{K}}

```

\Upmu
13520 \providecommand*\Upmu{\mathrm{M}}

\Upnu
13521 \providecommand*\Upnu{\mathrm{N}}

\Upomicron
13522 \providecommand*\Upomicron{\mathrm{O}}

\Uprho
13523 \providecommand*\Uprho{\mathrm{P}}

\Uptau
13524 \providecommand*\Uptau{\mathrm{T}}

\Upchi
13525 \providecommand*\Upchi{\mathrm{X}}

\upomicron
13526 \providecommand*\upomicron{\mathrm{o}}

13527}%
13528{}% 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.

```
13529 \newcommand*{\glxtrcontrolrules}{%
13530 \string'\glshex 200B\string'\string=\glshex 200C\string=\glshex 200D
13531 \string=\glshex 200E\string=\glshex 200F\string=\glshex 0000\string=\glshex 0001
13532 \string=\glshex 0002\string=\glshex 0003\string=\glshex 0004\string=\glshex 0005
13533 \string=\glshex 0006\string=\glshex 0007\string=\glshex 0008
13534 \string=\string'\glshex 0009\string'\string=\string'\glshex 000B\string'
13535 \string=\glshex 000E\string=\glshex 000F\string=\string'\glshex
13536 0010\string'\string=\glshex 0011
13537 \string=\glshex 0012\string=\glshex 0013\string=\glshex 0014\string=\glshex 0015
13538 \string=\glshex 0016\string=\glshex 0017\string=\glshex 0018\string=\glshex 0019
13539 \string=\glshex 001A\string=\glshex 001B\string=\glshex 001C\string=\glshex 001D
13540 \string=\glshex 001E\string=\glshex 001F\string=\glshex 007F\string=\glshex 0080
13541 \string=\glshex 0081\string=\glshex 0082\string=\glshex 0083\string=\glshex 0084
13542 \string=\glshex 0085\string=\glshex 0086\string=\glshex 0087\string=\glshex 0088
13543 \string=\glshex 0089\string=\glshex 008A\string=\glshex 008B\string=\glshex 008C
13544 \string=\glshex 008D\string=\glshex 008E\string=\glshex 008F\string=\glshex 0090
13545 \string=\glshex 0091\string=\glshex 0092\string=\glshex 0093\string=\glshex 0094
13546 \string=\glshex 0095\string=\glshex 0096\string=\glshex 0097\string=\glshex 0098
13547 \string=\glshex 0099\string=\glshex 009A\string=\glshex 009B\string=\glshex 009C
13548 \string=\glshex 009D\string=\glshex 009E\string=\glshex 009F
13549 }
```

lsxtrspacerules These are space characters.

```
13550 \newcommand*{\glxtrspacerules}{%
13551 \string' \string'\string;
13552 \string'\glshex 00A0\string'\string;
13553 \string'\glshex 2000\string'\string;
13554 \string'\glshex 2001\string'\string;
13555 \string'\glshex 2002\string'\string;
13556 \string'\glshex 2003\string'\string;
13557 \string'\glshex 2004\string'\string;
13558 \string'\glshex 2005\string'\string;
13559 \string'\glshex 2006\string'\string;
13560 \string'\glshex 2007\string'\string;
13561 \string'\glshex 2008\string'\string;
13562 \string'\glshex 2009\string'\string;
13563 \string'\glshex 200A\string'\string;
13564 \string'\glshex 3000\string'
13565 }
```

nprintablerules These are non-printable characters (BOM, tabs, line feed and carriage return).

```
13566 \newcommand*{\glxtrnonprintablerules}{%
13567 \string'\glshex FEFF\string'\string;
13568 \string'\glshex 000A\string'\string;
13569 \string'\glshex 0009\string'\string;
```



```

13570 \string'\glshex 000C\string'\string;
13571 \string'\glshex 000B\string'
13572 }

```

gdiacriticrules Combining diacritic marks. This is split into multiple macros.

```

13573 \newcommand*{\glxtrcombiningdiacriticrules}{%
13574 \glxtrcombiningdiacriticIrules\string;
13575 \glxtrcombiningdiacriticIIrules\string;
13576 \glxtrcombiningdiacriticIIIrules\string;
13577 \glxtrcombiningdiacriticIVrules
13578 }

```

diacriticIrules First set of combining diacritic marks.

```

13579 \newcommand*{\glxtrcombiningdiacriticIrules}{%
13580 \glshex 0301\string;% combining acute
13581 \glshex 0300\string;% combining grave
13582 \glshex 0306\string;% combining breve
13583 \glshex 0302\string;% combining circumflex
13584 \glshex 030C\string;% combining caron
13585 \glshex 030A\string;% combining ring
13586 \glshex 030D\string;% combining vertical line above
13587 \glshex 0308\string;% combining diaeresis
13588 \glshex 030B\string;% combining double acute
13589 \glshex 0303\string;% combining tilde
13590 \glshex 0307\string;% combining dot above
13591 \glshex 0304% combining macron
13592 }

```

diacriticIIrules Second set of combining diacritic marks.

```

13593 \newcommand*{\glxtrcombiningdiacriticIIrules}{%
13594 \glshex 0337\string;% combining short solidus overlay
13595 \glshex 0327\string;% combining cedilla
13596 \glshex 0328\string;% combining ogonek
13597 \glshex 0323\string;% combining dot below
13598 \glshex 0332\string;% combining low line
13599 \glshex 0305\string;% combining overline
13600 \glshex 0309\string;% combining hook above
13601 \glshex 030E\string;% combining double vertical line above
13602 \glshex 030F\string;% combining double grave accent
13603 \glshex 0310\string;% combining candrabindu
13604 \glshex 0311\string;% combining inverted breve
13605 \glshex 0312\string;% combining turned comma above
13606 \glshex 0313\string;% combining comma above
13607 \glshex 0314\string;% combining reversed comma above
13608 \glshex 0315\string;% combining comma above right
13609 \glshex 0316\string;% combining grave accent below
13610 \glshex 0317% combining acute accent below
13611 }

```

acriticIIIrules Third set of combining diacritic marks.

```
13612 \newcommand*{\glxtrcombiningdiacriticIIIrules}{%
13613 \glshex 0318\string;% combining left tack below
13614 \glshex 0319\string;% combining right tack below
13615 \glshex 031A\string;% combining left angle above
13616 \glshex 031B\string;% combining horn
13617 \glshex 031C\string;% combining left half ring below
13618 \glshex 031D\string;% combining up tack below
13619 \glshex 031E\string;% combining down tack below
13620 \glshex 031F\string;% combining plus sign below
13621 \glshex 0320\string;% combining minus sign below
13622 \glshex 0321\string;% combining palatalized hook below
13623 \glshex 0322\string;% combining retroflex hook below
13624 \glshex 0324\string;% combining diaeresis below
13625 \glshex 0325\string;% combining ring below
13626 \glshex 0326\string;% combining comma below
13627 \glshex 0329\string;% combining vertical line below
13628 \glshex 032A\string;% combining bridge below
13629 \glshex 032B\string;% combining inverted double arch below
13630 \glshex 032C\string;% combining caron below
13631 \glshex 032D\string;% combining circumflex accent below
13632 \glshex 032E\string;% combining breve below
13633 \glshex 032F\string;% combining inverted breve below
13634 \glshex 0330\string;% combining tilde below
13635 \glshex 0331\string;% combining macron below
13636 \glshex 0333\string;% combining double low line
13637 \glshex 0334\string;% combining tilde overlay
13638 \glshex 0335\string;% combining short stroke overlay
13639 \glshex 0336\string;% combining long stroke overlay
13640 \glshex 0338\string;% combining long solidus overlay
13641 \glshex 0339\string;% combining combining right half ring below
13642 \glshex 033A\string;% combining inverted bridge below
13643 \glshex 033B\string;% combining square below
13644 \glshex 033C\string;% combining seagull below
13645 \glshex 033D\string;% combining x above
13646 \glshex 033E\string;% combining vertical tilde
13647 \glshex 033F\string;% combining double overline
13648 \glshex 0342\string;% combining Greek perispomeni
13649 \glshex 0344\string;% combining Greek dialytika tonos
13650 \glshex 0345\string;% combining Greek ypogegrammeni
13651 \glshex 0360\string;% combining double tilde
13652 \glshex 0361\string;% combining double inverted breve
13653 \glshex 0483\string;% combining Cyrillic titlo
13654 \glshex 0484\string;% combining Cyrillic palatalization
13655 \glshex 0485\string;% combining Cyrillic dasia pneumata
13656 \glshex 0486% combining Cyrillic psili pneumata
13657 }
```

iacriticIVrules Fourth set of combining diacritic marks.

```

13658 \newcommand*{\glxtrcombiningdiacriticIVrules}{%
13659 \glshex 20D0\string;% combining left harpoon above
13660 \glshex 20D1\string;% combining right harpoon above
13661 \glshex 20D2\string;% combining long vertical line overlay
13662 \glshex 20D3\string;% combining short vertical line overlay
13663 \glshex 20D4\string;% combining anticlockwise arrow above
13664 \glshex 20D5\string;% combining clockwise arrow above
13665 \glshex 20D6\string;% combining left arrow above
13666 \glshex 20D7\string;% combining right arrow above
13667 \glshex 20D8\string;% combining ring overlay
13668 \glshex 20D9\string;% combining clockwise ring overlay
13669 \glshex 20DA\string;% combining anticlockwise ring overlay
13670 \glshex 20DB\string;% combining three dots above
13671 \glshex 20DC\string;% combining four dots above
13672 \glshex 20DD\string;% combining enclosing circle
13673 \glshex 20DE\string;% combining enclosing square
13674 \glshex 20DF\string;% combining enclosing diamond
13675 \glshex 20E0\string;% combining enclosing circle backslash
13676 \glshex 20E1% combining left right arrow above
13677 }

```

sxtrhyphenrules Hyphens.

```

13678 \newcommand*{\glxtrhyphenrules}{%
13679 \string'\string-\string'\string;% ASCII hyphen
13680 \glshex 00AD\string;% soft hyphen
13681 \glshex 2010\string;% hyphen
13682 \glshex 2011\string;% non-breaking hyphen
13683 \glshex 2012\string;% figure dash
13684 \glshex 2013\string;% en dash
13685 \glshex 2014\string;% em dash
13686 \glshex 2015\string;% horizontal bar
13687 \glshex 2212\string=\glshex 207B\string=\glshex 208B% minus sign
13688 }

```

eneralpuncrules General punctuation.

```

13689 \newcommand*{\glxtrgeneralpuncrules}{%
13690 \glxtrgeneralpuncIrules
13691 \string<\glxtrcurrencyrules
13692 \string<\glxtrgeneralpuncIIrules
13693 }

```

eneralpuncIrules First set of general punctuation.

```

13694 \newcommand*{\glxtrgeneralpuncIrules}{%
13695 \string'\glshex 005F\string'% underscore
13696 \string<\glshex 00AF% macron
13697 \string<\string'\glshex 002C\string'% comma
13698 \string<\string'\glshex 003B\string'% semi-colon
13699 \string<\string'\glshex 003A\string'% colon
13700 \string<\string'\glshex 0021\string'% exclamation mark

```

13701 \string<\glshex 00A1% inverted exclamation mark
 13702 \string<\string'\glshex 003F\string'% question mark
 13703 \string<\glshex 00BF% inverted question mark
 13704 \string<\string'\glshex 002F\string'% solidus
 13705 \string<\string'\glshex 002E\string'% full stop
 13706 \string<\glshex 00B4% acute accent
 13707 \string<\string'\glshex 0060\string'% grave accent
 13708 \string<\string'\glshex 005E\string'% circumflex accent
 13709 \string<\glshex 00A8% diaeresis
 13710 \string<\string'\glshex 007E\string'% tilde
 13711 \string<\glshex 00B7% middle dot
 13712 \string<\glshex 00B8% cedilla
 13713 \string<\string'\glshex 0027\string'% straight apostrophe
 13714 \string<\string'\glshex 0022\string'% straight double quote
 13715 \string<\glshex 00AB% left guillemet
 13716 \string<\glshex 00BB% right guillemet
 13717 \string<\string'\glshex 0028\string'% left parenthesis
 13718 \string=>\glshex 207D\string=>\glshex 208D% super/subscript left parenthesis
 13719 \string<\string'\glshex 0029\string'% right parenthesis
 13720 \string=>\glshex 207E\string=>\glshex 208E% super/subscript right parenthesis
 13721 \string<\string'\glshex 005B\string'% left square bracket
 13722 \string<\string'\glshex 005D\string'% right square bracket
 13723 \string<\string'\glshex 007B\string'% left curly bracket
 13724 \string<\string'\glshex 007D\string'% right curly bracket
 13725 \string<\glshex 00A7% section sign
 13726 \string<\glshex 00B6% pilcrow sign
 13727 \string<\glshex 00A9% copyright sign
 13728 \string<\glshex 00AE% registered sign
 13729 \string<\string'\glshex 0040\string'% at sign
 13730 }

trcurrencyrules General punctuation.

13731 \newcommand*{\glxtrcurrencyrules}{%
 13732 \glshex 00A4% currency sign
 13733 \string<\glshex 0E3F% Thai currency symbol baht
 13734 \string<\glshex 00A2% cent sign
 13735 \string<\glshex 20A1% colon sign
 13736 \string<\glshex 20A2% cruzeiro sign
 13737 \string<\string'\glshex 0024\string'% dollar sign
 13738 \string<\glshex 20AB% dong sign
 13739 \string<\glshex 20AC% euro sign
 13740 \string<\glshex 20A3% French franc sign
 13741 \string<\glshex 20A4% lira sign
 13742 \string<\glshex 20A5% mill sign
 13743 \string<\glshex 20A6% naira sign
 13744 \string<\glshex 20A7% peseta sign
 13745 \string<\glshex 00A3% pound sign
 13746 \string<\glshex 20A8% rupee sign
 13747 \string<\glshex 20AA% new sheqel sign

```

13748 \string<\glshex 20A9% won sign
13749 \string<\glshex 00A5% yen sign
13750 }

```

eralpuncIIrules Second set of general punctuation.

```

13751 \newcommand*{\glxtrgeneralpuncIIrules}{%
13752 \string'\glshex 002A\string'% asterisk
13753 \string<\string'\glshex 005C\string'% backslash
13754 \string<\string'\glshex 0026\string'% ampersand
13755 \string<\string'\glshex 0023\string'% hash sign
13756 \string<\string'\glshex 0025\string'% percent sign
13757 \string<\string'\glshex 002B\string'% plus sign
13758 \string=\glshex 207A\string=\glshex 208A% super/subscript plus sign
13759 \string<\glshex 00B1% plus-minus sign
13760 \string<\glshex 00F7% division sign
13761 \string<\glshex 00D7% multiplication sign
13762 \string<\string'\glshex 003C\string'% less-than sign
13763 \string<\string'\glshex 003D\string'% equals sign
13764 \string<\string'\glshex 003E\string'% greater-than sign
13765 \string<\glshex 00AC% not sign
13766 \string<\string'\glshex 007C\string'% vertical bar (pipe)
13767 \string<\glshex 00A6% broken bar
13768 \string<\glshex 00B0% degree sign
13769 \string<\glshex 00B5% micron sign
13770 }

```

eralLatinIrules Basic Latin alphabet.

```

13771 \newcommand*{\glxtrGeneralLatinIrules}{%
13772 \glxtrLatinA
13773 \string<b,B%
13774 \string<c,C%
13775 \string<d,D%
13776 \string<\glxtrLatinE
13777 \string<f,F%
13778 \string<g,G%
13779 \string<\glxtrLatinH
13780 \string<\glxtrLatinI
13781 \string<j,J%
13782 \string<\glxtrLatinK
13783 \string<\glxtrLatinL
13784 \string<\glxtrLatinM
13785 \string<\glxtrLatinN
13786 \string<\glxtrLatinO
13787 \string<\glxtrLatinP
13788 \string<q,Q%
13789 \string<r,R%
13790 \string<\glxtrLatinS
13791 \string<\glxtrLatinT
13792 \string<u,U%

```

```

13793 \string<v,V%
13794 \string<w,W%
13795 \string<\glxtrLatinX
13796 \string<y,Y%
13797 \string<z,Z
13798 }

```

allLatinIIrules General Latin alphabet (eth between D and E, ß treated as SS).

```

13799 \newcommand*{\glxtrGeneralLatinIIrules}{%
13800 \glxtrLatinA
13801 \string<b,B%
13802 \string<c,C%
13803 \string<d,D%
13804 \string<\glxtrLatinEth
13805 \string<\glxtrLatinE
13806 \string<f,F%
13807 \string<g,G%
13808 \string<\glxtrLatinH
13809 \string<\glxtrLatinI
13810 \string<j,J%
13811 \string<\glxtrLatinK
13812 \string<\glxtrLatinL
13813 \string<\glxtrLatinM
13814 \string<\glxtrLatinN
13815 \string<\glxtrLatinO
13816 \string<\glxtrLatinP
13817 \string<q,Q%
13818 \string<r,R%
13819 \string<\glxtrLatinS
13820 \string& SS \string, \glxtrLatinEszettSs
13821 \string<\glxtrLatinT
13822 \string<u,U%
13823 \string<v,V%
13824 \string<w,W%
13825 \string<\glxtrLatinX
13826 \string<y,Y%
13827 \string<z,Z%
13828 }

```

allLatinIIIrules General Latin alphabet (eth between D and E, ß treated as SZ).

```

13829 \newcommand*{\glxtrGeneralLatinIIIrules}{%
13830 \glxtrLatinA
13831 \string<b,B%
13832 \string<c,C%
13833 \string<d,D%
13834 \string<\glxtrLatinEth
13835 \string<\glxtrLatinE
13836 \string<f,F%
13837 \string<g,G%

```

```

13838 \string<\glxtrLatinH
13839 \string<\glxtrLatinI
13840 \string<j,J%
13841 \string<\glxtrLatinK
13842 \string<\glxtrLatinL
13843 \string<\glxtrLatinM
13844 \string<\glxtrLatinN
13845 \string<\glxtrLatinO
13846 \string<\glxtrLatinP
13847 \string<q,Q%
13848 \string<r,R%
13849 \string<\glxtrLatinS
13850 \string& SZ, \glxtrLatinEszettSz
13851 \string<\glxtrLatinT
13852 \string<u,U%
13853 \string<v,V%
13854 \string<w,W%
13855 \string<\glxtrLatinX
13856 \string<y,Y%
13857 \string<z,Z%
13858 }

```

`\glxtrGeneralLatinIVrules` General Latin alphabet (Æ treated as AE and Ætreated as OE, Þtreated as TH, ß treated as SS, eth between D and E).

```

13859 \newcommand*{\glxtrGeneralLatinIVrules}{%
13860 \glxtrLatinA
13861 \string& AE , \glxtrLatinAELigature
13862 \string<b,B%
13863 \string<c,C%
13864 \string<d,D%
13865 \string<\glxtrLatinEth
13866 \string<\glxtrLatinE
13867 \string<f,F%
13868 \string<g,G%
13869 \string<\glxtrLatinH
13870 \string<\glxtrLatinI
13871 \string<j,J%
13872 \string<\glxtrLatinK
13873 \string<\glxtrLatinL
13874 \string<\glxtrLatinM
13875 \string<\glxtrLatinN
13876 \string<\glxtrLatinO
13877 \string& OE , \glxtrLatinOELigature
13878 \string<\glxtrLatinP
13879 \string<q,Q%
13880 \string<r,R%
13881 \string<\glxtrLatinS
13882 \string& SS , \glxtrLatinEszettSs
13883 \string<\glxtrLatinT

```

```

13884 \string& th =\glshex 00DE
13885 \string& TH =\glshex 00FE
13886 \string<u,U%
13887 \string<v,V%
13888 \string<w,W%
13889 \string<\glxtrLatinX
13890 \string<y,Y%
13891 \string<z,Z%
13892 }

```

eralLatinVrules General Latin alphabet (eth between D and E, ß treated as SS, Þ treated as TH).

```

13893 \newcommand*{\glxtrGeneralLatinVrules}{%
13894 \glxtrLatinA
13895 \string<b,B%
13896 \string<c,C%
13897 \string<d,D%
13898 \string<\glxtrLatinEth
13899 \string<\glxtrLatinE
13900 \string<f,F%
13901 \string<g,G%
13902 \string<\glxtrLatinH
13903 \string<\glxtrLatinI
13904 \string<j,J%
13905 \string<\glxtrLatinK
13906 \string<\glxtrLatinL
13907 \string<\glxtrLatinM
13908 \string<\glxtrLatinN
13909 \string<\glxtrLatinO
13910 \string<\glxtrLatinP
13911 \string<q,Q%
13912 \string<r,R%
13913 \string<\glxtrLatinS
13914 \string& SS , \glxtrLatinEszettSs
13915 \string<\glxtrLatinT
13916 \string& th =\glshex 00DE
13917 \string& TH =\glshex 00FE
13918 \string<u,U%
13919 \string<v,V%
13920 \string<w,W%
13921 \string<\glxtrLatinX
13922 \string<y,Y%
13923 \string<z,Z%
13924 }

```

ralLatinVrules General Latin alphabet (eth between D and E, ß treated as SZ, Þ treated as TH).

```

13925 \newcommand*{\glxtrGeneralLatinVrules}{%
13926 \glxtrLatinA
13927 \string<b,B%
13928 \string<c,C%

```



```

13929 \string<d,D%
13930 \string<\glxtrLatinEth
13931 \string<\glxtrLatinE
13932 \string<f,F%
13933 \string<g,G%
13934 \string<\glxtrLatinH
13935 \string<\glxtrLatinI
13936 \string<j,J%
13937 \string<\glxtrLatinK
13938 \string<\glxtrLatinL
13939 \string<\glxtrLatinM
13940 \string<\glxtrLatinN
13941 \string<\glxtrLatinO
13942 \string<\glxtrLatinP
13943 \string<q,Q%
13944 \string<r,R%
13945 \string<\glxtrLatinS
13946 \string& SZ , \glxtrLatinEszettSz
13947 \string<\glxtrLatinT
13948 \string& th =\glshex 00DE
13949 \string& TH =\glshex 00FE
13950 \string<u,U%
13951 \string<v,V%
13952 \string<w,W%
13953 \string<\glxtrLatinX
13954 \string<y,Y%
13955 \string<z,Z%
13956 }

```

allLatinVIIrules General Latin alphabet (Æ between A and B, eth between D and E, insular G as G, Œ between O and P, long S equivalent to S, Þ between T and U and wynn as W).

```

13957 \newcommand*{\glxtrGeneralLatinVIIrules}{%
13958 \glxtrLatinA
13959 \string<\glxtrLatinAELigature
13960 \string<b,B%
13961 \string<c,C%
13962 \string<d,D%
13963 \string<\glxtrLatinEth
13964 \string<\glxtrLatinE
13965 \string<f,F%
13966 \string<\glxtrLatinInsularG
13967 \string<\glxtrLatinH
13968 \string<\glxtrLatinI
13969 \string<j,J%
13970 \string<\glxtrLatinK
13971 \string<\glxtrLatinL
13972 \string<\glxtrLatinM
13973 \string<\glxtrLatinN
13974 \string<\glxtrLatinO

```

```

13975 \string<\glxtrLatinOELigature
13976 \string<\glxtrLatinP
13977 \string<q,Q%
13978 \string<r,R%
13979 \string<\glshex 017F=\glxtrLatinS % s and long s
13980 \string<\glxtrLatinT
13981 \string<\glxtrLatinThorn
13982 \string<u,U%
13983 \string<v,V%
13984 \string< w\string=\glshex 01BF, W\string=\glshex 01F7
13985 \string<\glxtrLatinX
13986 \string<y,Y%
13987 \string<z,Z%
13988 }

```

1LatinVIIIrules 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).

```

13989 \newcommand*{\glxtrGeneralLatinVIIIrules}{%
13990 \glxtrLatinA
13991 \string& AE , \glxtrLatinAELigature
13992 \string<b,B%
13993 \string<c,C%
13994 \string<\glshex 00F0\string;d,\glshex 00D0\string;D% D and eth
13995 \string<\glxtrLatinE
13996 \string<f,F%
13997 \string<g,G%
13998 \string<\glxtrLatinH
13999 \string<\glxtrLatinI
14000 \string<j,J%
14001 \string<\glxtrLatinK
14002 \string<\glshex 0142\string=\glxtrLatinL\string=\glshex 0141% L and \L
14003 \string<\glxtrLatinM
14004 \string<\glxtrLatinN
14005 \string<\glshex 00F8\string=\glxtrLatinO\string=\glshex 00D8% O and \O
14006 \string& OE , \glxtrLatinOELigature
14007 \string<\glxtrLatinP
14008 \string<q,Q%
14009 \string<r,R%
14010 \string<\glxtrLatinS
14011 \string& SS , \glxtrLatinEszettSs
14012 \string<\glxtrLatinT
14013 \string& th =\glshex 00DE
14014 \string& TH =\glshex 00FE
14015 \string<u,U%
14016 \string<v,V%
14017 \string<w,W%
14018 \string<\glxtrLatinX
14019 \string<y,Y%
14020 \string<z,Z%

```

14021 }

\glsxtrLatinA

14022 \newcommand*{\glsxtrLatinA}{%

14023 a\string=\glshex 00AA\string=\glshex 2090,A

14024 }

\glsxtrLatinE

14025 \newcommand*{\glsxtrLatinE}{%

14026 e\string=\glshex 2091,E

14027 }

\glsxtrLatinH

14028 \newcommand*{\glsxtrLatinH}{%

14029 h\string=\glshex 2095,H

14030 }

\glsxtrLatinI

14031 \newcommand*{\glsxtrLatinI}{%

14032 i\string=\glshex 2071,I

14033 }

\glsxtrLatinK

14034 \newcommand*{\glsxtrLatinK}{%

14035 k\string=\glshex 2096,K

14036 }

\glsxtrLatinL

14037 \newcommand*{\glsxtrLatinL}{%

14038 l\string=\glshex 2097,L

14039 }

\glsxtrLatinM

14040 \newcommand*{\glsxtrLatinM}{%

14041 m\string=\glshex 2098,M

14042 }

\glsxtrLatinN

14043 \newcommand*{\glsxtrLatinN}{%

14044 n\string=\glshex 207F\string=\glshex 2099,N

14045 }

\glsxtrLatinO

14046 \newcommand*{\glsxtrLatinO}{%

14047 o\string=\glshex 00BA\string=\glshex 2092,O

14048 }

```

\glxtrLatinP
14049 \newcommand*{\glxtrLatinP}{%
14050   p\string=\glshex 209A,P
14051 }

\glxtrLatinS
14052 \newcommand*{\glxtrLatinS}{%
14053   s\string=\glshex 209B,S
14054 }

\glxtrLatinT
14055 \newcommand*{\glxtrLatinT}{%
14056   t\string=\glshex 209C,T
14057 }

\glxtrLatinX
14058 \newcommand*{\glxtrLatinX}{%
14059   x\string=\glshex 2093,X
14060 }

lsxtrLatinSchwa  Latin schwa (lower case, subscript and upper case).
14061 \newcommand*{\glxtrLatinSchwa}{%
14062   \glshex 0259\string=\glshex 2094,\glshex 018F
14063 }

trLatinEszettSs
14064 \newcommand*{\glxtrLatinEszettSs}{%
14065   \glshex 00DF% eszett
14066   \string=\glshex 017Fs % long S s
14067 }

trLatinEszettSz
14068 \newcommand*{\glxtrLatinEszettSz}{%
14069   \glshex 00DF% eszett
14070   \string= \glshex 017Fz % long S z
14071 }

\glxtrLatinEth
14072 \newcommand*{\glxtrLatinEth}{%
14073   \glshex 00F0,\glshex 00D0% eth
14074 }

lsxtrLatinThorn
14075 \newcommand*{\glxtrLatinThorn}{%
14076   \glshex 00FE,\glshex 00DE% thorn
14077 }

```

LatinAELigature

```
14078 \newcommand*{\glxtrLatinAELigature}{%
14079 \glshex 00E6,\glshex 00C6% AE-ligature
14080 }
```

LatinOELigature

```
14081 \newcommand*{\glxtrLatinOELigature}{%
14082 \glshex 0153,\glshex 0152% OE-ligature
14083 }
```

\glxtrLatinAA

```
14084 \newcommand*{\glxtrLatinAA}{%
14085 \glshex 00E5=a\glshex 030A,% \aa
14086 \glshex 00C5=A\glshex 030A% \AA
14087 }
```

glxtrLatinWynn

```
14088 \newcommand*{\glxtrLatinWynn}{%
14089 \glshex 01BF,\glshex 01F7% wynn
14090 }
```

trLatinInsularG

```
14091 \newcommand*{\glxtrLatinInsularG}{%
14092 \glshex 1D79,\glshex A77D% insular G
14093 \string; g, G
14094 }
```

sxtrLatinOslash

```
14095 \newcommand*{\glxtrLatinOslash}{%
14096 \glshex 00F8,\glshex 00D8% \o, \O
14097 }
```

sxtrLatinLslash

```
14098 \newcommand*{\glxtrLatinLslash}{%
14099 \glshex 0142,\glshex 0141% \l, \L
14100 }
```

thUpGreekIrules Includes digamma between epsilon and zeta.

```
14101 \newcommand*{\glxtrMathUpGreekIrules}{%
14102 \glxtrUpAlpha
14103 \string<\glxtrUpBeta
14104 \string<\glxtrUpGamma
14105 \string<\glxtrUpDelta
14106 \string<\glxtrUpEpsilon
14107 \string<\glxtrUpDigamma
14108 \string<\glxtrUpZeta
14109 \string<\glxtrUpEta
14110 \string<\glxtrUpTheta
```

```

14111 \string<\glxtrUpIota
14112 \string<\glxtrUpKappa
14113 \string<\glxtrUpLambda
14114 \string<\glxtrUpMu
14115 \string<\glxtrUpNu
14116 \string<\glxtrUpXi
14117 \string<\glxtrUpOmicron
14118 \string<\glxtrUpPi
14119 \string<\glxtrUpRho
14120 \string<\glxtrUpSigma
14121 \string<\glxtrUpTau
14122 \string<\glxtrUpUpsilon
14123 \string<\glxtrUpPhi
14124 \string<\glxtrUpChi
14125 \string<\glxtrUpPsi
14126 \string<\glxtrUpOmega
14127 }

```

hUpGreekIIrules Doesn't include digamma.

```

14128 \newcommand*{\glxtrMathUpGreekIIrules}{%
14129 \glxtrUpAlpha
14130 \string<\glxtrUpBeta
14131 \string<\glxtrUpGamma
14132 \string<\glxtrUpDelta
14133 \string<\glxtrUpEpsilon
14134 \string<\glxtrUpZeta
14135 \string<\glxtrUpEta
14136 \string<\glxtrUpTheta
14137 \string<\glxtrUpIota
14138 \string<\glxtrUpKappa
14139 \string<\glxtrUpLambda
14140 \string<\glxtrUpMu
14141 \string<\glxtrUpNu
14142 \string<\glxtrUpXi
14143 \string<\glxtrUpOmicron
14144 \string<\glxtrUpPi
14145 \string<\glxtrUpRho
14146 \string<\glxtrUpSigma
14147 \string<\glxtrUpTau
14148 \string<\glxtrUpUpsilon
14149 \string<\glxtrUpPhi
14150 \string<\glxtrUpChi
14151 \string<\glxtrUpPsi
14152 \string<\glxtrUpOmega
14153 }

```

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.

```

14154 \newcommand*{\glxtrMathItalicGreekIrules}{%

```

```

14155 \glxtrMathItalicAlpha
14156 \string<\glxtrMathItalicBeta
14157 \string<\glxtrMathItalicGamma
14158 \string<\glxtrMathItalicDelta
14159 \string<\glxtrMathItalicEpsilon
14160 \string<\glxtrUpDigamma
14161 \string<\glxtrMathItalicZeta
14162 \string<\glxtrMathItalicEta
14163 \string<\glxtrMathItalicTheta
14164 \string<\glxtrMathItalicIota
14165 \string<\glxtrMathItalicKappa
14166 \string<\glxtrMathItalicLambda
14167 \string<\glxtrMathItalicMu
14168 \string<\glxtrMathItalicNu
14169 \string<\glxtrMathItalicXi
14170 \string<\glxtrMathItalicOmicron
14171 \string<\glxtrMathItalicPi
14172 \string<\glxtrMathItalicRho
14173 \string<\glxtrMathItalicSigma
14174 \string<\glxtrMathItalicTau
14175 \string<\glxtrMathItalicUpsilon
14176 \string<\glxtrMathItalicPhi
14177 \string<\glxtrMathItalicChi
14178 \string<\glxtrMathItalicPsi
14179 \string<\glxtrMathItalicOmega
14180 }

```

licGreekIIrules Doesn't include digamma.

```

14181 \newcommand*{\glxtrMathItalicGreekIIrules}{%
14182 \glxtrMathItalicAlpha
14183 \string<\glxtrMathItalicBeta
14184 \string<\glxtrMathItalicGamma
14185 \string<\glxtrMathItalicDelta
14186 \string<\glxtrMathItalicEpsilon
14187 \string<\glxtrMathItalicZeta
14188 \string<\glxtrMathItalicEta
14189 \string<\glxtrMathItalicTheta
14190 \string<\glxtrMathItalicIota
14191 \string<\glxtrMathItalicKappa
14192 \string<\glxtrMathItalicLambda
14193 \string<\glxtrMathItalicMu
14194 \string<\glxtrMathItalicNu
14195 \string<\glxtrMathItalicXi
14196 \string<\glxtrMathItalicOmicron
14197 \string<\glxtrMathItalicPi
14198 \string<\glxtrMathItalicRho
14199 \string<\glxtrMathItalicSigma
14200 \string<\glxtrMathItalicTau
14201 \string<\glxtrMathItalicUpsilon

```

```

14202 \string<\glxtrMathItalicPhi
14203 \string<\glxtrMathItalicChi
14204 \string<\glxtrMathItalicPsi
14205 \string<\glxtrMathItalicOmega
14206 }

```

UpperGreekIrules Upper case only (includes upright digamma).

```

14207 \newcommand*{\glxtrMathItalicUpperGreekIrules}{%
14208 \glshex 1D6E2% upper case alpha (maths italic)
14209 \string<\glshex 1D6E3% upper case beta (maths italic)
14210 \string<\glshex 1D6E4% upper case gamma (maths italic)
14211 \string<\glshex 1D6E5% upper case delta (maths italic)
14212 \string<\glshex 1D6E6% upper case epsilon (maths italic)
14213 \string<\glshex 03DC% upper case digamma
14214 \string<\glshex 1D6E7% upper case zeta (maths italic)
14215 \string<\glshex 1D6E8% upper case eta (maths italic)
14216 \string<\glshex 1D6E9% upper case theta (maths italic)
14217 \string<\glshex 1D6F3% upper case theta variant (maths italic)
14218 \string<\glshex 1D6EA% upper case iota (maths italic)
14219 \string<\glshex 1D6EB% upper case kappa (maths italic)
14220 \string<\glshex 1D6EC% upper case lambda (maths italic)
14221 \string<\glshex 1D6ED% upper case mu (maths italic)
14222 \string<\glshex 1D6EE% upper case nu (maths italic)
14223 \string<\glshex 1D6EF% upper case xi (maths italic)
14224 \string<\glshex 1D6F0% upper case omicron (maths italic)
14225 \string<\glshex 1D6F1% upper case pi (maths italic)
14226 \string<\glshex 1D6F2% upper case rho (maths italic)
14227 \string<\glshex 1D6F4% upper case sigma (maths italic)
14228 \string<\glshex 1D6F5% upper case tau (maths italic)
14229 \string<\glshex 1D6F6% upper case upsilon (maths italic)
14230 \string<\glshex 1D6F7% upper case phi (maths italic)
14231 \string<\glshex 1D6F8% upper case chi (maths italic)
14232 \string<\glshex 1D6F9% upper case psi (maths italic)
14233 \string<\glshex 1D6FA% upper case omega (maths italic)
14234 }

```

UpperGreekIIrules Upper case only (doesn't include upright digamma).

```

14235 \newcommand*{\glxtrMathItalicUpperGreekIIrules}{%
14236 \glshex 1D6E2% upper case alpha (maths italic)
14237 \string<\glshex 1D6E3% upper case beta (maths italic)
14238 \string<\glshex 1D6E4% upper case gamma (maths italic)
14239 \string<\glshex 1D6E5% upper case delta (maths italic)
14240 \string<\glshex 1D6E6% upper case epsilon (maths italic)
14241 \string<\glshex 1D6E7% upper case zeta (maths italic)
14242 \string<\glshex 1D6E8% upper case eta (maths italic)
14243 \string<\glshex 1D6E9% upper case theta (maths italic)
14244 \string<\glshex 1D6F3% upper case theta variant (maths italic)
14245 \string<\glshex 1D6EA% upper case iota (maths italic)
14246 \string<\glshex 1D6EB% upper case kappa (maths italic)

```



```

14247 \string<\glshex 1D6EC% upper case lambda (maths italic)
14248 \string<\glshex 1D6ED% upper case mu (maths italic)
14249 \string<\glshex 1D6EE% upper case nu (maths italic)
14250 \string<\glshex 1D6EF% upper case xi (maths italic)
14251 \string<\glshex 1D6F0% upper case omicron (maths italic)
14252 \string<\glshex 1D6F1% upper case pi (maths italic)
14253 \string<\glshex 1D6F2% upper case rho (maths italic)
14254 \string<\glshex 1D6F4% upper case sigma (maths italic)
14255 \string<\glshex 1D6F5% upper case tau (maths italic)
14256 \string<\glshex 1D6F6% upper case upsilon (maths italic)
14257 \string<\glshex 1D6F7% upper case phi (maths italic)
14258 \string<\glshex 1D6F8% upper case chi (maths italic)
14259 \string<\glshex 1D6F9% upper case psi (maths italic)
14260 \string<\glshex 1D6FA% upper case omega (maths italic)
14261 }

```

LowerGreekIrules Lower case only (includes upright digamma).

```

14262 \newcommand*{\glstrMathItalicLowerGreekIrules}{%
14263 \glshex 1D6FC% lower case alpha (maths italic)
14264 \string<\glshex 1D6FD% lower case beta (maths italic)
14265 \string<\glshex 1D6FE% lower case gamma (maths italic)
14266 \string<\glshex 1D6FF% lower case delta (maths italic)
14267 \string<\glshex 1D700% lower case epsilon (maths italic)
14268 \string=<\glshex 1D716% lower case epsilon variant (maths italic)
14269 \string<\glshex 03DD% lower case digamma
14270 \string<\glshex 1D701% lower case zeta (maths italic)
14271 \string<\glshex 1D702% lower case eta (maths italic)
14272 \string<\glshex 1D703% lower case theta (maths italic)
14273 \string=<\glshex 1D717% lower case theta variant (maths italic)
14274 \string<\glshex 1D704% lower case iota (maths italic)
14275 \string<\glshex 1D705% lower case kappa (maths italic)
14276 \string=<\glshex 1D718% lower case kappa variant (maths italic)
14277 \string<\glshex 1D706% lower case lambda (maths italic)
14278 \string<\glshex 1D707% lower case mu (maths italic)
14279 \string<\glshex 1D708% lower case nu (maths italic)
14280 \string<\glshex 1D709% lower case xi (maths italic)
14281 \string<\glshex 1D70A% lower case omicron (maths italic)
14282 \string<\glshex 1D70B% lower case pi (maths italic)
14283 \string=<\glshex 1D71B% lower case pi variant (maths italic)
14284 \string<\glshex 1D70C% lower case rho (maths italic)
14285 \string=<\glshex 1D71A% lower case rho variant (maths italic)
14286 \string<\glshex 1D70D% lower case final sigma (maths italic)
14287 \string=<\glshex 1D70E% lower case sigma (maths italic)
14288 \string<\glshex 1D70F% lower case tau (maths italic)
14289 \string<\glshex 1D710% lower case upsilon (maths italic)
14290 \string<\glshex 1D711% lower case phi (maths italic)
14291 \string=<\glshex 1D719% lower case phi variant (maths italic)
14292 \string<\glshex 1D712% lower case chi (maths italic)
14293 \string<\glshex 1D713% lower case psi (maths italic)

```

```

14294 \string<\glshex 1D714% lower case omega (maths italic)
14295 }

```

LowerGreeklIrules Lower case only (doesn't includes upright digamma).

```

14296 \newcommand*{\glxtrMathItalicLowerGreeklIrules}{%
14297 \glshex 1D6FC% lower case alpha (maths italic)
14298 \string<\glshex 1D6FD% lower case beta (maths italic)
14299 \string<\glshex 1D6FE% lower case gamma (maths italic)
14300 \string<\glshex 1D6FF% lower case delta (maths italic)
14301 \string<\glshex 1D700% lower case epsilon (maths italic)
14302 \string=>\glshex 1D716% lower case epsilon variant (maths italic)
14303 \string<\glshex 1D701% lower case zeta (maths italic)
14304 \string<\glshex 1D702% lower case eta (maths italic)
14305 \string<\glshex 1D703% lower case theta (maths italic)
14306 \string=>\glshex 1D717% lower case theta variant (maths italic)
14307 \string<\glshex 1D704% lower case iota (maths italic)
14308 \string<\glshex 1D705% lower case kappa (maths italic)
14309 \string=>\glshex 1D718% lower case kappa variant (maths italic)
14310 \string<\glshex 1D706% lower case lambda (maths italic)
14311 \string<\glshex 1D707% lower case mu (maths italic)
14312 \string<\glshex 1D708% lower case nu (maths italic)
14313 \string<\glshex 1D709% lower case xi (maths italic)
14314 \string<\glshex 1D70A% lower case omicron (maths italic)
14315 \string<\glshex 1D70B% lower case pi (maths italic)
14316 \string=>\glshex 1D71B% lower case pi variant (maths italic)
14317 \string<\glshex 1D70C% lower case rho (maths italic)
14318 \string=>\glshex 1D71A% lower case rho variant (maths italic)
14319 \string<\glshex 1D70D% lower case final sigma (maths italic)
14320 \string=>\glshex 1D70E% lower case sigma (maths italic)
14321 \string<\glshex 1D70F% lower case tau (maths italic)
14322 \string<\glshex 1D710% lower case upsilon (maths italic)
14323 \string<\glshex 1D711% lower case phi (maths italic)
14324 \string=>\glshex 1D719% lower case phi variant (maths italic)
14325 \string<\glshex 1D712% lower case chi (maths italic)
14326 \string<\glshex 1D713% lower case psi (maths italic)
14327 \string<\glshex 1D714% lower case omega (maths italic)
14328 }

```

MathGreeklIrules Includes both upright and italic with digamma between epsilon and zeta.

```

14329 \newcommand*{\glxtrMathGreeklIrules}{%
14330 \glxtrMathItalicAlpha
14331 \string;\glxtrUpAlpha
14332 \string<\glxtrMathItalicBeta
14333 \string;\glxtrUpBeta
14334 \string<\glxtrMathItalicGamma
14335 \string;\glxtrUpGamma
14336 \string<\glxtrMathItalicDelta
14337 \string;\glxtrUpDelta
14338 \string<\glxtrMathItalicEpsilon

```

```

14339 \string;\glxtrUpEpsilon
14340 \string<\glxtrUpDigamma
14341 \string<\glxtrMathItalicZeta
14342 \string;\glxtrUpZeta
14343 \string<\glxtrMathItalicEta
14344 \string;\glxtrUpEta
14345 \string<\glxtrMathItalicTheta
14346 \string;\glxtrUpTheta
14347 \string<\glxtrMathItalicIota
14348 \string;\glxtrUpIota
14349 \string<\glxtrMathItalicKappa
14350 \string;\glxtrUpKappa
14351 \string<\glxtrMathItalicLambda
14352 \string;\glxtrUpLambda
14353 \string<\glxtrMathItalicMu
14354 \string;\glxtrUpMu
14355 \string<\glxtrMathItalicNu
14356 \string;\glxtrUpNu
14357 \string<\glxtrMathItalicXi
14358 \string;\glxtrUpXi
14359 \string<\glxtrMathItalicOmicron
14360 \string;\glxtrUpOmicron
14361 \string<\glxtrMathItalicPi
14362 \string;\glxtrUpPi
14363 \string<\glxtrMathItalicRho
14364 \string;\glxtrUpRho
14365 \string<\glxtrMathItalicSigma
14366 \string;\glxtrUpSigma
14367 \string<\glxtrMathItalicTau
14368 \string;\glxtrUpTau
14369 \string<\glxtrMathItalicUpsilon
14370 \string;\glxtrUpUpsilon
14371 \string<\glxtrMathItalicPhi
14372 \string;\glxtrUpPhi
14373 \string<\glxtrMathItalicChi
14374 \string;\glxtrUpChi
14375 \string<\glxtrMathItalicPsi
14376 \string;\glxtrUpPsi
14377 \string<\glxtrMathItalicOmega
14378 \string;\glxtrUpOmega
14379 }

```

mathGreekIIrules Includes both upright and italic (digamma not included).

```

14380 \newcommand*{\glxtrMathGreekIIrules}{%
14381 \glxtrMathItalicAlpha
14382 \string;\glxtrUpAlpha
14383 \string<\glxtrMathItalicBeta
14384 \string;\glxtrUpBeta
14385 \string<\glxtrMathItalicGamma

```

```

14386 \string;\glsxtrUpGamma
14387 \string<\glsxtrMathItalicDelta
14388 \string;\glsxtrUpDelta
14389 \string<\glsxtrMathItalicEpsilon
14390 \string;\glsxtrUpEpsilon
14391 \string<\glsxtrMathItalicZeta
14392 \string;\glsxtrUpZeta
14393 \string<\glsxtrMathItalicEta
14394 \string;\glsxtrUpEta
14395 \string<\glsxtrMathItalicTheta
14396 \string;\glsxtrUpTheta
14397 \string<\glsxtrMathItalicIota
14398 \string;\glsxtrUpIota
14399 \string<\glsxtrMathItalicKappa
14400 \string;\glsxtrUpKappa
14401 \string<\glsxtrMathItalicLambda
14402 \string;\glsxtrUpLambda
14403 \string<\glsxtrMathItalicMu
14404 \string;\glsxtrUpMu
14405 \string<\glsxtrMathItalicNu
14406 \string;\glsxtrUpNu
14407 \string<\glsxtrMathItalicXi
14408 \string;\glsxtrUpXi
14409 \string<\glsxtrMathItalicOmicron
14410 \string;\glsxtrUpOmicron
14411 \string<\glsxtrMathItalicPi
14412 \string;\glsxtrUpPi
14413 \string<\glsxtrMathItalicRho
14414 \string;\glsxtrUpRho
14415 \string<\glsxtrMathItalicSigma
14416 \string;\glsxtrUpSigma
14417 \string<\glsxtrMathItalicTau
14418 \string;\glsxtrUpTau
14419 \string<\glsxtrMathItalicUpsilon
14420 \string;\glsxtrUpUpsilon
14421 \string<\glsxtrMathItalicPhi
14422 \string;\glsxtrUpPhi
14423 \string<\glsxtrMathItalicChi
14424 \string;\glsxtrUpChi
14425 \string<\glsxtrMathItalicPsi
14426 \string;\glsxtrUpPsi
14427 \string<\glsxtrMathItalicOmega
14428 \string;\glsxtrUpOmega
14429 }

```

`\glsxtrUpAlpha`

```

14430 \newcommand*{\glsxtrUpAlpha}{%
14431 \glshex 03B1,% lower case alpha
14432 \glshex 0391% upper case alpha

```

14433 }

\glsxtrUpBeta

14434 \newcommand*{\glsxtrUpBeta}{%
14435 \glshex 03B2,% lower case beta
14436 \glshex 0392% upper case beta
14437 }

\glsxtrUpGamma

14438 \newcommand*{\glsxtrUpGamma}{%
14439 \glshex 03B3,% lower case gamma
14440 \glshex 0393% upper case gamma
14441 }

\glsxtrUpDelta

14442 \newcommand*{\glsxtrUpDelta}{%
14443 \glshex 03B4,% lower case delta
14444 \glshex 0394% upper case delta
14445 }

glsxtrUpEpsilon

14446 \newcommand*{\glsxtrUpEpsilon}{%
14447 \glshex 03B5% lower case epsilon
14448 \string=\glshex 03F5,% lower case epsilon variant
14449 \glshex 0395% upper case epsilon
14450 }

glsxtrUpDigamma

14451 \newcommand*{\glsxtrUpDigamma}{%
14452 \glshex 03DD,% lower case digamma
14453 \glshex 03DC% upper case digamma
14454 }

\glsxtrUpZeta

14455 \newcommand*{\glsxtrUpZeta}{%
14456 \glshex 03B6,% lower case zeta
14457 \glshex 0396% upper case zeta
14458 }

\glsxtrUpEta

14459 \newcommand*{\glsxtrUpEta}{%
14460 \glshex 03B7,% lower case eta
14461 \glshex 0397% upper case eta
14462 }

\glsxtrUpTheta

14463 \newcommand*{\glsxtrUpTheta}{%
14464 \glshex 03B8% lower case theta

```

14465 \string=\glshex 03D1,% lower case theta variant
14466 \glshex 0398% upper case theta
14467 }

```

\glxtrUpIota

```

14468 \newcommand*{\glxtrUpIota}{%
14469 \glshex 03B9,% lower case iota
14470 \glshex 0399% upper case iota
14471 }

```

\glxtrUpKappa

```

14472 \newcommand*{\glxtrUpKappa}{%
14473 \glshex 03BA,% lower case kappa
14474 \string=\glshex 03F0,% lower case kappa variant
14475 \glshex 039A% upper case kappa
14476 }

```

\glxtrUpLambda

```

14477 \newcommand*{\glxtrUpLambda}{%
14478 \glshex 03BB,% lower lambda
14479 \glshex 039B% upper case lambda
14480 }

```

\glxtrUpMu

```

14481 \newcommand*{\glxtrUpMu}{%
14482 \glshex 03BC,% lower case mu
14483 \glshex 039C% upper case mu
14484 }

```

\glxtrUpNu

```

14485 \newcommand*{\glxtrUpNu}{%
14486 \glshex 03BD,% lower case nu
14487 \glshex 039D% upper case nu
14488 }

```

\glxtrUpXi

```

14489 \newcommand*{\glxtrUpXi}{%
14490 \glshex 03BE,% lower case xi
14491 \glshex 039E% upper case xi
14492 }

```

\glxtrUpOmicron

```

14493 \newcommand*{\glxtrUpOmicron}{%
14494 \glshex 03BF,% lower case omicron
14495 \glshex 039F% upper case omicron
14496 }

```

\glxtrUpPi

```
14497 \newcommand*{\glxtrUpPi}{%
14498   \glshex 03C0% lower case pi
14499   \string=\glshex 03D6,% lower case pi variant
14500   \glshex 03A0% upper case pi
14501 }
```

\glxtrUpRho

```
14502 \newcommand*{\glxtrUpRho}{%
14503   \glshex 03C1% lower case rho
14504   \string=\glshex 03F1,% lower case rho variant
14505   \glshex 03A1% upper case rho
14506 }
```

\glxtrUpSigma

```
14507 \newcommand*{\glxtrUpSigma}{%
14508   \glshex 03C2% lower case sigma
14509   \string=\glshex 03C3,% lower case sigma
14510   \glshex 03A3% upper case sigma
14511 }
```

\glxtrUpTau

```
14512 \newcommand*{\glxtrUpTau}{%
14513   \glshex 03C4,% lower case tau
14514   \glshex 03A4% upper case tau
14515 }
```

\glxtrUpUpsilon

```
14516 \newcommand*{\glxtrUpUpsilon}{%
14517   \glshex 03C5,% lower case upsilon
14518   \glshex 03A5% upper case upsilon
14519 }
```

\glxtrUpPhi

```
14520 \newcommand*{\glxtrUpPhi}{%
14521   \glshex 03C6% lower case phi
14522   \string=\glshex 03D5,% lower case phi variant
14523   \glshex 03A6% upper case phi
14524 }
```

\glxtrUpChi

```
14525 \newcommand*{\glxtrUpChi}{%
14526   \glshex 03C7,% lower case chi
14527   \glshex 03A7% upper case chi
14528 }
```

\glxtrUpPsi

```
14529 \newcommand*{\glxtrUpPsi}{%
```

```

14530 \glshex 03C8,% lower case psi
14531 \glshex 03A8% upper case psi
14532 }

```

$\backslash\mathrm{glxtrUpOmega}$

```

14533 \newcommand*{\glxtrUpOmega}{%
14534 \glshex 03C9,% lower case omega
14535 \glshex 03A9% upper case omega
14536 }

```

$\mathrm{MathItalicAlpha}$

```

14537 \newcommand*{\glxtrMathItalicAlpha}{%
14538 \glshex 1D6FC,% lower case alpha (maths italic)
14539 \glshex 1D6E2% upper case alpha (maths italic)
14540 }

```

$\mathrm{rMathItalicBeta}$

```

14541 \newcommand*{\glxtrMathItalicBeta}{%
14542 \glshex 1D6FD,% lower case beta (maths italic)
14543 \glshex 1D6E3% upper case beta (maths italic)
14544 }

```

$\mathrm{MathItalicGamma}$

```

14545 \newcommand*{\glxtrMathItalicGamma}{%
14546 \glshex 1D6FE,% lower case gamma (maths italic)
14547 \glshex 1D6E4% upper case gamma (maths italic)
14548 }

```

$\mathrm{MathItalicDelta}$

```

14549 \newcommand*{\glxtrMathItalicDelta}{%
14550 \glshex 1D6FF,% lower case delta (maths italic)
14551 \glshex 1D6E5% upper case delta (maths italic)
14552 }

```

$\mathrm{thItalicEpsilon}$

```

14553 \newcommand*{\glxtrMathItalicEpsilon}{%
14554 \glshex 1D700% lower case epsilon (maths italic)
14555 \string=\glshex 1D716,% lower case epsilon variant (maths italic)
14556 \glshex 1D6E6% upper case epsilon (maths italic)
14557 }

```

$\mathrm{rMathItalicZeta}$

```

14558 \newcommand*{\glxtrMathItalicZeta}{%
14559 \glshex 1D701,% lower case zeta (maths italic)
14560 \glshex 1D6E7% upper case zeta (maths italic)
14561 }

```


trMathItalicEta

```
14562 \newcommand*{\glxtrMathItalicEta}{%
14563   \glshex 1D702,% lower case eta (maths italic)
14564   \glshex 1D6E8% upper case eta (maths italic)
14565 }
```

MathItalicTheta

```
14566 \newcommand*{\glxtrMathItalicTheta}{%
14567   \glshex 1D703% lower case theta (maths italic)
14568   \string=\glshex 1D717,% lower case theta variant (maths italic)
14569   \glshex 1D6E9% upper case theta (maths italic)
14570   \string=\glshex 1D6F3% upper case theta variant (maths italic)
14571 }
```

rMathItalicIota

```
14572 \newcommand*{\glxtrMathItalicIota}{%
14573   \glshex 1D704,% lower case iota (maths italic)
14574   \glshex 1D6EA% upper case iota (maths italic)
14575 }
```

MathItalicKappa

```
14576 \newcommand*{\glxtrMathItalicKappa}{%
14577   \glshex 1D705% lower case kappa (maths italic)
14578   \string=\glshex 1D718,% lower case kappa variant (maths italic)
14579   \glshex 1D6EB% upper case kappa (maths italic)
14580 }
```

athItalicLambda

```
14581 \newcommand*{\glxtrMathItalicLambda}{%
14582   \glshex 1D706,% lower case lambda (maths italic)
14583   \glshex 1D6EC% upper case lambda (maths italic)
14584 }
```

xtrMathItalicMu

```
14585 \newcommand*{\glxtrMathItalicMu}{%
14586   \glshex 1D707,% lower case mu (maths italic)
14587   \glshex 1D6ED% upper case mu (maths italic)
14588 }
```

xtrMathItalicNu

```
14589 \newcommand*{\glxtrMathItalicNu}{%
14590   \glshex 1D708,% lower case nu (maths italic)
14591   \glshex 1D6EE% upper case nu (maths italic)
14592 }
```

xtrMathItalicXi

```
14593 \newcommand*{\glxtrMathItalicXi}{%
14594   \glshex 1D709,% lower case xi (maths italic)
```

```

14595 \glshex 1D6EF% upper case xi (maths italic)
14596 }

thItalicOmicron

14597 \newcommand*{\glxtrMathItalicOmicron}{%
14598 \glshex 1D70A,% lower case omicron (maths italic)
14599 \glshex 1D6F0% upper case omicron (maths italic)
14600 }

xtrMathItalicPi

14601 \newcommand*{\glxtrMathItalicPi}{%
14602 \glshex 1D70B,% lower case pi (maths italic)
14603 \string=\glshex 1D71B,% lower case pi variant (maths italic)
14604 \glshex 1D6F1% upper case pi (maths italic)
14605 }

trMathItalicRho

14606 \newcommand*{\glxtrMathItalicRho}{%
14607 \glshex 1D70C,% lower case rho (maths italic)
14608 \string=\glshex 1D71A,% lower case rho variant (maths italic)
14609 \glshex 1D6F2% upper case rho (maths italic)
14610 }

MathItalicSigma

14611 \newcommand*{\glxtrMathItalicSigma}{%
14612 \glshex 1D70D,% lower case final sigma (maths italic)
14613 \string=\glshex 1D70E,% lower case sigma (maths italic)
14614 \glshex 1D6F4% upper case sigma (maths italic)
14615 }

trMathItalicTau

14616 \newcommand*{\glxtrMathItalicTau}{%
14617 \glshex 1D70F,% lower case tau (maths italic)
14618 \glshex 1D6F5% upper case tau (maths italic)
14619 }

thItalicUpsilon

14620 \newcommand*{\glxtrMathItalicUpsilon}{%
14621 \glshex 1D710,% lower case upsilon (maths italic)
14622 \glshex 1D6F6% upper case upsilon (maths italic)
14623 }

trMathItalicPhi

14624 \newcommand*{\glxtrMathItalicPhi}{%
14625 \glshex 1D711,% lower case phi (maths italic)
14626 \string=\glshex 1D719,% lower case phi variant (maths italic)
14627 \glshex 1D6F7% upper case phi (maths italic)
14628 }

```

trMathItalicChi

```
14629 \newcommand*{\glxtrMathItalicChi}{%
14630   \glshex 1D712,% lower case chi (maths italic)
14631   \glshex 1D6F8% upper case chi (maths italic)
14632 }
```

trMathItalicPsi

```
14633 \newcommand*{\glxtrMathItalicPsi}{%
14634   \glshex 1D713,% lower case psi (maths italic)
14635   \glshex 1D6F9% upper case psi (maths italic)
14636 }
```

MathItalicOmega

```
14637 \newcommand*{\glxtrMathItalicOmega}{%
14638   \glshex 1D714,% lower case omega (maths italic)
14639   \glshex 1D6FA% upper case omega (maths italic)
14640 }
```

thItalicPartial

```
14641 \newcommand*{\glxtrMathItalicPartial}{%
14642   \glshex 1D715% partial differential (maths italic)
14643 }
```

MathItalicNabla

```
14644 \newcommand*{\glxtrMathItalicNabla}{%
14645   \glshex 1D6FB% nabla (maths italic)
14646 }
```

lsxtrdigitrules Digits from the Basic Latin set and subscript and superscript digit rules.

```
14647 \newcommand*{\glxtrdigitrules}{%
14648   0\string=\glshex 2080\string=\glshex 2070
14649   \string<1\string=\glshex 2081\string=\glshex 00B9
14650   \string<2\string=\glshex 2082\string=\glshex 00B2
14651   \string<3\string=\glshex 2083\string=\glshex 00B3
14652   \string<4\string=\glshex 2084\string=\glshex 2074
14653   \string<5\string=\glshex 2085\string=\glshex 2075
14654   \string<6\string=\glshex 2086\string=\glshex 2076
14655   \string<7\string=\glshex 2087\string=\glshex 2077
14656   \string<8\string=\glshex 2088\string=\glshex 2078
14657   \string<9\string=\glshex 2089\string=\glshex 2079
14658 }
```

BasicDigitrules Digits from the Basic Latin set.

```
14659 \newcommand*{\glxtrBasicDigitrules}{%
14660   0\string<1\string<2\string<3\string<4%
14661   \string<5\string<6\string<7\string<8\string<9%
14662 }
```

criptDigitrules Subscript digits.

```
14663 \newcommand*{\glxtrSubScriptDigitrules}{%
14664 \glshex 2080% subscript 0
14665 \string<\glshex 2081% subscript 1
14666 \string<\glshex 2082% subscript 2
14667 \string<\glshex 2083% subscript 3
14668 \string<\glshex 2084% subscript 4
14669 \string<\glshex 2085% subscript 5
14670 \string<\glshex 2086% subscript 6
14671 \string<\glshex 2087% subscript 7
14672 \string<\glshex 2088% subscript 8
14673 \string<\glshex 2089% subscript 9
14674 }
```

criptDigitrules Superscript digits.

```
14675 \newcommand*{\glxtrSuperScriptDigitrules}{%
14676 \glshex 2070% superscript 0
14677 \string<\glshex 00B9% superscript 1
14678 \string<\glshex 00B2% superscript 2
14679 \string<\glshex 00B3% superscript 3
14680 \string<\glshex 2074% superscript 4
14681 \string<\glshex 2075% superscript 5
14682 \string<\glshex 2076% superscript 6
14683 \string<\glshex 2077% superscript 7
14684 \string<\glshex 2078% superscript 8
14685 \string<\glshex 2079% superscript 9
14686 }
```

trfractionrules Vulgar fractions.

```
14687 \newcommand*{\glxtrfractionrules}{%
14688 \glshex 215F% fraction numerator one (1/)
14689 \string<\glshex 2189% zero thirds (0/3 = 0)
14690 \string<\glshex 2152% one tenth (1/10 = 0.1)
14691 \string<\glshex 2151% one ninth (1/9 ~ 0.111)
14692 \string<\glshex 215B% one eighth (1/8 = 0.125)
14693 \string<\glshex 2150% one seventh (1/7 ~ 0.143)
14694 \string<\glshex 2159% one sixth (1/6 ~ 0.167)
14695 \string<\glshex 2155% one fifth (1/5 = 0.2)
14696 \string<\glshex 00BC% one quarter (1/4 = 0.25)
14697 \string<\glshex 2153% one third (1/3 ~ 0.333)
14698 \string<\glshex 215C% three eighths (3/8 = 0.375)
14699 \string<\glshex 2156% two fifths (2/5 = 0.4)
14700 \string<\glshex 00BD% one half (1/2 = 0.5)
14701 \string<\glshex 2157% three fifths (3/5 = 0.6)
14702 \string<\glshex 215D% five eighths (5/8 = 0.625)
14703 \string<\glshex 2154% two thirds (2/3 ~ 0.667)
14704 \string<\glshex 00BE% three quarters (3/4 = 0.75)
14705 \string<\glshex 2158% four fifths (4/5 = 0.8)
14706 \string<\glshex 215A% five sixths (5/6 ~ 0.833)
```

```

14707 \string<\glshex 215E% seven eighths (7/8 = 0.875)
14708 }

```

sxtrdialecthook Check for scripts associated with the document dialects.

```

14709 \renewcommand{\@glsxtrdialecthook}{%
14710   \ifundef\CurrentTrackedScript
14711   {%
14712     \TrackLangIfHasDefaultScript{\CurrentTrackedLanguage}%
14713     {%
14714       \edef\CurrentTrackedScript{%
14715         \TrackLangGetDefaultScript\CurrentTrackedLanguage}%
14716     }%
14717   }%
14718 }%
14719 {}%
14720 \ifdef\CurrentTrackedScript
14721 {%
14722   \let\gls@orgTrackLangRequireDialectPrefix\TrackLangRequireDialectPrefix
14723   \def\TrackLangRequireDialectPrefix{glossariesxtr-}%
14724   \let\CurrentTrackedTag\CurrentTrackedScript
14725   \IfFileExists{\TrackLangRequireDialectPrefix\CurrentTrackedTag.ldf}
14726   {\RequireGlossariesExtraLang{\CurrentTrackedTag}}%
14727   {}%
14728   \let\TrackLangRequireDialectPrefix\gls@orgTrackLangRequireDialectPrefix
14729 }%
14730 {}%
14731 }

```

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.

```

14732 \ifdef\glsxtr@loaddialect
14733 {%
14734   \@ifpackageloaded{tracklang}
14735   {%
14736     \AnyTrackedLanguages
14737     {%
14738       \ForEachTrackedDialect{\this@dialect}{\glsxtr@loaddialect}%
14739     }%
14740   }%
14741 }
14742 {}
14743 }
14744 {}

```

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:

```
14745 \NeedsTeXFormat{LaTeX2e}
14746 \ProvidesPackage{glossaries-extra-stylemods}[2021/09/20 v1.46 (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`

```
14747 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
14748 \DeclareOption{all}{%
14749   \appto\@glxtr@loadstyles{%
14750     \RequirePackage{glossary-inline}%
14751     \RequirePackage{glossary-list}%
14752     \RequirePackage{glossary-tree}%
14753     \RequirePackage{glossary-mcols}%
14754     \RequirePackage{glossary-long}%
14755     \RequirePackage{glossary-longragged}%
14756     \RequirePackage{glossary-longbooktabs}%
14757     \RequirePackage{glossary-super}%
14758     \RequirePackage{glossary-superragged}%
14759     \RequirePackage{glossary-bookindex}%
14760     \RequirePackage{glossary-longextra}%
14761     \RequirePackage{glossary-topic}%
14762   }
14763 }

14764 \DeclareOption*{%
14765   \IfFileExists{glossary-\CurrentOption.sty}
14766   {\eappto\@glxtr@loadstyles{%
14767     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
14768   }%
```

```

14769   {%
14770     \PackageError{glossaries-extra-styles}%
14771     {Unknown option ‘\CurrentOption’}{}%
14772   }%
14773 }

```

Process the package options:

```
14774 \ProcessOptions
```

Load the required packages:

```
14775 \@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.

```
14776 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

ewglossarystyle

```

14777 \providecommand{\renewglossarystyle}[2]{%
14778   \ifcsundef{@glsstyle@#1}%
14779   {%
14780     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
14781   }%
14782   {%
14783     \csdef{@glsstyle@#1}{#2}%
14784   }%
14785 }

```

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.

```

14786 \ifdef{\@glsstyle@listdotted}
14787 {%
14788   \renewglossarystyle{listdotted}{%
14789     \setglossarystyle{list}%
14790     \renewcommand*{\glossentry}[2]{%
14791       \item[]\makebox[\glslistdottedwidth][l]{%
14792         \glsentryitem{##1}%
14793         \glstarget{##1}{\glossentryname{##1}}%
14794         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
14795         \glossentrydesc{##1}\glspostdescription}%
14796     \renewcommand*{\subglossentry}[3]{%
14797       \item[]\makebox[\glslistdottedwidth][l]{%
14798         \glssubentryitem{##2}%
14799         \glstarget{##2}{\glossentryname{##2}}%
14800         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%

```

```

14801 \glossentrydesc{##2}\glspostdescription}%
14802 }
14803 }
14804 {%

```

Assume the style isn't required if it hasn't already been defined.

```

14805 }

```

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.

```

14806 \ifdef{\@glsstyle@list}
14807 {%

```

listprelocation Space before number list for top-level entries.

```

14808 \newcommand{\glslistprelocation}{\glstrprelocation}

```

childprelocation Space before number list for child entries.

```

14809 \newcommand{\glslistchildprelocation}{\glslistprelocation}

```

childpostlocation Full stop after number list.

```

14810 \newcommand{\glslistchildpostlocation}{.}

```

\glslistdesc

```

14811 \newcommand{\glslistdesc}[1]{\glossentrydesc{#1}\glspostdescription}

```

lslistgroupskip

```

14812 \newcommand{\glslistgroupskip}{\nobreak\indexspace\nobreak}

```

Redefine list to use these commands.

```

14813 \renewglossarystyle{list}{%
14814 \renewenvironment{theglossary}%
14815 {\begin{description}}{\end{description}}%
14816 \renewcommand*\{glossaryheader}{}%
14817 \renewcommand*\{glsgroupheading}[1]{}%
14818 \renewcommand*\{glossentry}[2]{%
14819 \item[\glssentryitem{##1}%
14820 \glstarget{##1}{\glossentryname{##1}}]
14821 \glslistdesc{##1}\glslistprelocation ##2}%
14822 \renewcommand*\{subglossentry}[3]{%
14823 \glssubentryitem{##2}%
14824 \glstarget{##2}{\strut}\space
14825 \glslistdesc{##2}%
14826 \glslistchildprelocation ##3\glslistchildpostlocation}%
14827 \renewcommand*\{glsgroupskip}{\ifglsnogroupskip\else\glslistgroupskip\fi}%
14828 }
14829 }
14830 {}

```


Similarly for altlist. Since it requires list, the new commands should have been defined above.

```

14831 \ifdef{\@glsstyle@altlist}
14832 {%
14833   \renewglossarystyle{altlist}{%
14834     \setglossarystyle{list}%
14835     \renewcommand*{\glossentry}[2]{%
14836       \item[\glsentryitem{##1}]%
14837       \glstarget{##1}{\glossentryname{##1}}}%
14838     \mbox{}\par\nobreak\@afterheading
14839     \glslistdesc{##1}\glslistprelocation ##2}%
14840   \renewcommand{\subglossentry}[3]{%
14841     \par
14842     \glssubentryitem{##2}%
14843     \glstarget{##2}{\strut}\glslistdesc{##2}%
14844     \glslistchildprelocation ##3}%
14845   }
14846 }
14847 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

14848 \ifdef{\@glsstyle@listgroup}
14849 {%
14850   \renewglossarystyle{listgroup}{%
14851     \setglossarystyle{list}%
14852     \renewcommand*{\glsgroupheading}[1]{%
14853       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}]%
14854       \mbox{}\par\nobreak\@afterheading
14855     }%
14856   }
14857 }
14858 {}

```

Similarly for listhypergroup.

```

14859 \ifdef{\@glsstyle@listhypergroup}
14860 {%
14861   \renewglossarystyle{listhypergroup}{%
14862     \setglossarystyle{list}%
14863     \renewcommand*{\glossaryheader}{%
14864       \glslistnavigationitem{\glsnavigation}}%
14865     \renewcommand*{\glsgroupheading}[1]{%
14866       \item[\glslistgroupheaderfmt
14867         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
14868       \mbox{}\par\nobreak\@afterheading
14869     }%
14870   }
14871 }
14872 {}

```

Similarly for altlistgroup.

```

14873 \ifdef{\@glsstyle@altlistgroup}

```

```

14874 {%
14875   \renewglossarystyle{altlistgroup}{%
14876     \setglossarystyle{altlist}%
14877     \renewcommand*{\glsgroupheading}[1]{%
14878       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}]%
14879       \mbox{}\par\nobreak\@afterheading
14880     }%
14881   }
14882 }
14883 {}

```

Similarly for altlisthypergroup.

```

14884 \ifdef{\@glsstyle@altlisthypergroup}
14885 {%
14886   \renewglossarystyle{altlisthypergroup}{%
14887     \setglossarystyle{altlist}%
14888     \renewcommand*{\glossaryheader}{%
14889       \glslistnavigationitem{\glsnavigation}}%
14890     \renewcommand*{\glsgroupheading}[1]{%
14891       \item[\glslistgroupheaderfmt
14892         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
14893       \mbox{}\par\nobreak\@afterheading
14894     }%
14895   }
14896 }
14897 {}

```

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 `\glxtrprelocation`.

```

14898 \ifcsdef{@glsstyle@long}
14899 {%
14900   \renewglossarystyle{long}{%
14901     \renewenvironment{theglossary}%
14902       {\begin{longtable}{lp{\glsdescwidth}}}%
14903       {\end{longtable}}%
14904     \renewcommand*{\glossaryheader}{}%
14905     \renewcommand*{\glsgroupheading}[1]{}%
14906     \renewcommand{\glossentry}[2]{%
14907       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
14908       \glossentrydesc{##1}\glspostdescription
14909       \glxtrprelocation ##2\tabularnewline
14910     }%
14911     \renewcommand{\subglossentry}[3]{%
14912       &
14913       \glssubentryitem{##2}%
14914       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription

```

```

14915      \glstrprelocation ##3\tabularnewline
14916    }%
14917    \ifglsgroupskip
14918      \renewcommand*{\glsgroupskip}{}%
14919    \else
14920      \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
14921    \fi
14922  }
14923 }
14924 {}

```

Three column style:

```

14925 \ifcsdef{@glstyle@long3col}
14926 {%
14927   \renewglossarystyle{long3col}{%
14928     \renewenvironment{theglossary}%
14929       {\begin{longtable}\lp{\glsgdescwidth}p{\glspagelistwidth}}}%
14930     {\end{longtable}}}%
14931   \renewcommand*{\glossaryheader}{}%
14932   \renewcommand*{\glsgroupheading}[1]{}%
14933   \renewcommand{\glossentry}[2]{%
14934     \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
14935     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
14936   }%
14937   \renewcommand{\subglossentry}[3]{%
14938     &
14939     \glssubentryitem{##2}%
14940     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
14941     ##3\tabularnewline
14942   }%

```

Conditional needs to be outside of `\glsgroupskip` otherwise it can cause “Incomplete `\iftrue`” errors.

```

14943   \ifglsgroupskip
14944     \renewcommand*{\glsgroupskip}{}%
14945   \else
14946     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
14947   \fi
14948 }
14949 }
14950 {}

```

Four column style:

```

14951 \ifcsdef{@glstyle@long4col}
14952 {%
14953   \renewglossarystyle{long4col}{%
14954     \renewenvironment{theglossary}%
14955       {\begin{longtable}\l{llll}}}%
14956     {\end{longtable}}}%
14957   \renewcommand*{\glossaryheader}{}%
14958   \renewcommand*{\glsgroupheading}[1]{}%

```

```

14959 \renewcommand{\glossentry}[2]{%
14960   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
14961   \glossentrydesc{##1}\glspostdescription &
14962   \glossentrysymbol{##1} &
14963   ##2\tabularnewline
14964 }%
14965 \renewcommand{\subglossentry}[3]{%
14966   &
14967   \glssubentryitem{##2}%
14968   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
14969   \glossentrysymbol{##2} & ##3\tabularnewline
14970 }%

14971 \ifglsnogroupskip
14972   \renewcommand*\{\glsgroupskip}{}%
14973 \else
14974   \renewcommand*\{\glsgroupskip}{& & \tabularnewline}%
14975 \fi
14976 }
14977 }
14978 {}

```

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`.

```

14979 \ifcsdef{@glstyle@longragged}
14980 {%
14981   \renewglossarystyle{longragged}{%
14982     \renewenvironment{theglossary}%
14983       {\begin{longtable}{l>{\raggedright}p{\glstdescwidth}}}%
14984       {\end{longtable}}%
14985     \renewcommand*\{\glossaryheader}{}%
14986     \renewcommand*\{\glsgroupheading}[1]{}%
14987     \renewcommand{\glossentry}[2]{%
14988       \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
14989       \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
14990       \tabularnewline
14991     }%
14992     \renewcommand{\subglossentry}[3]{%
14993       &
14994       \glssubentryitem{##2}%
14995       \glstarget{##2}{\strut}\glossentrydesc{##2}%
14996       \glspostdescription\glxtrprelocation ##3%
14997       \tabularnewline

```

```

14998 }%
14999 \ifglsgroupskip
15000 \renewcommand*{\glsgroupskip}{}%
15001 \else
15002 \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
15003 \fi
15004 }
15005 }
15006 {}

```

Three and four column styles don't use `\glstrprelocation` since the number list is in its own column.

```

15007 \ifcsdef{@glstyle@longragged3col}
15008 {%
15009 \renewglossarystyle{longragged3col}{%
15010 \renewenvironment{theglossary}%
15011 {\begin{longtable}{l>\raggedright}p{\glsglwidth}%
15012 >\raggedright}p{\glspagelistwidth}}}%
15013 {\end{longtable}}}%
15014 \renewcommand*{\glossaryheader}{}%
15015 \renewcommand*{\glsgroupheading}[1]{}%
15016 \renewcommand{\glossentry}[2]{%
15017 \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15018 \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
15019 }%
15020 \renewcommand{\subglossentry}[3]{%
15021 &
15022 \glssubentryitem{##2}%
15023 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15024 ##3\tabularnewline
15025 }%
15026 \ifglsgroupskip
15027 \renewcommand*{\glsgroupskip}{}%
15028 \else
15029 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
15030 \fi
15031 }
15032 }
15033 {}

```

Four column style:

```

15034 \ifcsdef{@glstyle@altlongragged4col}
15035 {%
15036 \renewglossarystyle{altlongragged4col}{%
15037 \renewenvironment{theglossary}%
15038 {\begin{longtable}{l>\raggedright}p{\glsglwidth}l%
15039 >\raggedright}p{\glspagelistwidth}}}%
15040 {\end{longtable}}}%
15041 \renewcommand*{\glossaryheader}{}%

```

```

15042 \renewcommand*\glsgroupheading}[1]{}%
15043 \renewcommand{\glossentry}[2]{%
15044     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15045     \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
15046     ##2\tabularnewline
15047 }%
15048 \renewcommand{\subglossentry}[3]{%
15049     &
15050     \glssubentryitem{##2}%
15051     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15052     \glossentrysymbol{##2} & ##3\tabularnewline
15053 }%

15054 \ifglsnogroupskip
15055     \renewcommand*\glsgroupskip{}{}%
15056 \else
15057     \renewcommand*\glsgroupskip{& & \tabularnewline}%
15058 \fi
15059 }
15060 }
15061 {}

```

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 \glstrprelocation.

```

15062 \ifcsdef{@glstyle@super}
15063 {%
15064     \renewglossarystyle{super}{%
15065         \renewenvironment{theglossary}%
15066             {\tablehead{}\tabletail{}}%
15067             \begin{supertabular}{\lp{\glstdescwidth}}}%
15068             {\end{supertabular}}%
15069     \renewcommand*\glossaryheader{}{}%
15070     \renewcommand*\glsgroupheading}[1]{}%
15071     \renewcommand{\glossentry}[2]{%
15072         \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15073         \glossentrydesc{##1}\glspostdescription
15074         \glstrprelocation ##2\tabularnewline
15075     }%
15076     \renewcommand{\subglossentry}[3]{%
15077         &
15078         \glssubentryitem{##2}%
15079         \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
15080         \glstrprelocation ##3\tabularnewline
15081     }%
15082     \ifglsnogroupskip
15083         \renewcommand*\glsgroupskip{}{}%

```

```

15084 \else
15085     \renewcommand*{\glsgroupskip}{& \tabularnewline}%
15086 \fi
15087 }
15088 }
15089 {}

```

Three column style:

```

15090 \ifcsdef{@glsstyle@super3col}
15091 {%
15092 \renewglossarystyle{super3col}{%
15093 \renewenvironment{theglossary}%
15094 {\tablehead{}\tabletail}%
15095 \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
15096 {\end{supertabular}}%
15097 \renewcommand*{\glossaryheader}{}%
15098 \renewcommand*{\glsgroupheading}[1]{}%
15099 \renewcommand{\glossentry}[2]{%
15100 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15101 \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
15102 }%
15103 \renewcommand{\subglossentry}[3]{%
15104 &
15105 \glssubentryitem{##2}%
15106 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15107 ##3\tabularnewline
15108 }%

15109 \ifglsnogroupskip
15110 \renewcommand*{\glsgroupskip}{}%
15111 \else
15112 \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
15113 \fi
15114 }
15115 }
15116 {}

```

Four column styles:

```

15117 \ifcsdef{@glsstyle@super4col}
15118 {%
15119 \renewglossarystyle{super4col}{%
15120 \renewenvironment{theglossary}%
15121 {\tablehead{}\tabletail}%
15122 \begin{supertabular}{llll}}{%
15123 \end{supertabular}}%
15124 \renewcommand*{\glossaryheader}{}%
15125 \renewcommand*{\glsgroupheading}[1]{}%
15126 \renewcommand{\glossentry}[2]{%
15127 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15128 \glossentrydesc{##1}\glspostdescription &

```

```

15129     \glossentrysymbol{##1} & ##2\tabularnewline
15130 }%
15131 \renewcommand{\subglossentry}[3]{%
15132     &
15133     \glssubentryitem{##2}%
15134     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15135     \glossentrysymbol{##2} & ##3\tabularnewline
15136 }%

15137 \ifglsnogroupskip
15138     \renewcommand*{\glsgroupskip}{}%
15139 \else
15140     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
15141 \fi
15142 }
15143 }
15144 {}

```

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 `\glxtrprelocation`.

```

15145 \ifcsdef{@glstyle@superragged}
15146 {%
15147     \renewglossarystyle{superragged}{%
15148         \renewenvironment{theglossary}%
15149             {\tablehead{}\tabletail}%
15150             \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}}}%
15151             {\end{supertabular}}%
15152     \renewcommand*{\glossaryheader}{}%
15153     \renewcommand*{\glsgroupheading}[1]{}%
15154     \renewcommand{\glossentry}[2]{%
15155         \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15156         \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
15157         \tabularnewline
15158     }%
15159     \renewcommand{\subglossentry}[3]{%
15160         &
15161         \glssubentryitem{##2}%
15162         \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
15163         \glxtrprelocation ##3%
15164         \tabularnewline
15165     }%
15166     \ifglsnogroupskip
15167         \renewcommand*{\glsgroupskip}{}%
15168     \else
15169         \renewcommand*{\glsgroupskip}{& \tabularnewline}%

```



```

15170 \fi
15171 }
15172 }
15173 {}

```

Three column style:

```

15174 \ifcsdef{@glsstyle@superragged3col}
15175 {%
15176 \renewglossarystyle{superragged3col}{%
15177 \renewenvironment{theglossary}%
15178 {\tablehead{}}\tabletail{}}%
15179 \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}%
15180 >{\raggedright}p{\glspagelistwidth}}}%
15181 {\end{supertabular}}%
15182 \renewcommand*{\glossaryheader}{}%
15183 \renewcommand*{\glsgroupheading}[1]{}%
15184 \renewcommand{\glossentry}[2]{%
15185 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15186 \glossentrydesc{##1}\glspostdescription &
15187 ##2\tabularnewline
15188 }%
15189 \renewcommand{\subglossentry}[3]{%
15190 &
15191 \glssubentryitem{##2}%
15192 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15193 ##3\tabularnewline
15194 }%
15195 \ifglsnogroupskip
15196 \renewcommand*{\glsgroupskip}{}%
15197 \else
15198 \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
15199 \fi
15200 }
15201 }
15202 {}

```

Four columns:

```

15203 \ifcsdef{@glsstyle@altsuperragged4col}
15204 {%
15205 \renewglossarystyle{altsuperragged4col}{%
15206 \renewenvironment{theglossary}%
15207 {\tablehead{}}\tabletail{}}%
15208 \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}l%
15209 >{\raggedright}p{\glspagelistwidth}}}%
15210 {\end{supertabular}}%
15211 \renewcommand*{\glossaryheader}{}%
15212 \renewcommand{\glossentry}[2]{%
15213 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15214 \glossentrydesc{##1}\glspostdescription &

```

```

15215     \glossentrysymbol{##1} & ##2\tabularnewline
15216 }%
15217 \renewcommand{\subglossentry}[3]{%
15218     &
15219     \glssubentryitem{##2}%
15220     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15221     \glossentrysymbol{##2} & ##3\tabularnewline
15222 }%

15223 \ifglsnogroupskip
15224     \renewcommand*{\glsgroupskip}{}%
15225 \else
15226     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
15227 \fi
15228 }
15229 }
15230 {}

```

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`.

```

15231 \ifdef{\@glsstyle@inline}
15232 {%
15233     \renewcommand*{\glspostinline}{.\spacefactor\sfcode{.}}

```

Just use `\glstrpostdescription` instead of `\glspostdescription`.

```

15234     \renewcommand*{\glsinlinedescformat}[3]{%
15235         \space#1\glstrpostdescription}
15236     \renewcommand*{\glsinlinesubdescformat}[3]{%
15237         #1\glstrpostdescription}

```

The default settings don't show the location lists, so there's no adjustment for `\glstrprelocation`.

```

15238 }
15239 {}

```

2.8 Tree Styles

Redefine both `\glstreenamfmt` and `\glstreegroupheaderfmt` in terms of `\glstreedefaultnamfmt` to make it easier to change both at the same time or only change one without affecting the other.

```

15240 \ifdef\glstreenamfmt
15241 {

```

`\glstreedefaultnamfmt`

```

15242     \newcommand{\glstreedefaultnamfmt}[1]{\textbf{#1}}

```

`\glstreenamefmt`

```
15243 \renewcommand{\glstreenamefmt}[1]{\glstreedefaultnamefmt{#1}}
```

`\glstreegroupheaderfmt` This command was only introduced to glossary-tree v4.22, so it may not be defined.

```
15244 \def\glstreegroupheaderfmt#1{\glstreedefaultnamefmt{#1}}
```

`\glstreenavigationfmt` This command was only introduced to glossary-tree v4.22, so it may not be defined.

```
15245 \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.

```
15246 \newcommand{\glstreePreHeader}[2]{}%
```

```
15247 }
```

```
15248 {}%
```

The index style is redefined so that the space before the number list isn't hard coded.

```
15249 \ifdef{\@glsstyle@index}
```

```
15250 {
```

`\glstreeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```
15251 \newcommand*{\glstreeprelocation}{\glstxtrprelocation}
```

`\glstreechildprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```
15252 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}
```

Don't prohibit a page break at the start of a new group if there's no header.

`\glstreegroupskip`

```
15253 \newcommand{\glstreegroupskip}{\indexspace}
```

`\glstreegroupheaderskip` This doesn't include `\@afterheading` as it can cause interference with some styles.

```
15254 \newcommand{\glstreegroupheaderskip}{\nopagebreak\glstreegroupskip\nobreak}
```

Modify the index style.

```
15255 \renewglossarystyle{index}{%
15256   \renewenvironment{theglossary}%
15257     {\setlength{\parindent}{0pt}}%
15258     {\setlength{\parskip}{0pt plus 0.3pt}}%
15259     \let\item\glstreeitem
15260     \let\subitem\glstreesubitem
15261     \let\subsubitem\glstreesubsubitem
15262   }%
15263   {\par}%
15264   \renewcommand*{\glossaryheader}{}%
15265   \renewcommand*{\glsgroupheading}[1]{}%
15266   \renewcommand*{\glossentry}[2]{%
15267     \item\glsentryitem{#1}%
```

```

15268     \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
15269     \glstreesymbol{##1}%
15270     \glstreeDescLoc{##1}{##2}%
15271 }%
15272 \renewcommand{\subglossentry}[3]{%
15273     \ifcase##1\relax
15274     \item
15275     \or
15276     \subitem
15277     \glssubentryitem{##2}%
15278     \else
15279     \subsubitem
15280     \fi
15281     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
15282     \glstreechildsymbol{##2}%
15283     \glstreeChildDescLoc{##2}{##3}%
15284 }%
15285 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\glstreegroupskip\fi}%
15286 }
15287 }
15288 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

15289 \ifdef{\@glsstyle@indexgroup}
15290 {%
15291     \renewglossarystyle{indexgroup}{%
15292         \setglossarystyle{index}%
15293         \renewcommand*{\glsgroupheading}[1]{%
15294             \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
15295             \glstreePreHeader{##1}{\glsxtr@grptitle}%
15296             \item\glstreegroupheaderfmt{\glsxtr@grptitle}%
15297             \glstreegroupheaderskip\@afterheading
15298         }%
15299     }
15300 }
15301 {}

```

Similarly for indexhypergroup.

```

15302 \ifdef{\@glsstyle@indexhypergroup}
15303 {%
15304     \renewglossarystyle{indexhypergroup}{%
15305         \setglossarystyle{index}%
15306         \renewcommand*{\glossaryheader}{%
15307             \item\glstreenavigationfmt{\glsnavigation}%
15308             \glstreegroupheaderskip\@afterheading}%
15309         \renewcommand*{\glsgroupheading}[1]{%
15310             \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
15311             \glstreePreHeader{##1}{\glsxtr@grptitle}%
15312             \item\glstreegroupheaderfmt
15313                 {\glsnahypertarget{##1}{\glsxtr@grptitle}}%

```

```

15314 \glstreegroupheaderskip\@afterheading}%
15315 }%
15316 }
15317 {}

```

Adjust tree style to remove hard coded space before number list.

```

15318 \ifdef{\@glstyle@tree}
15319 {%

```

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

```

15320 \newcommand{\glsxtrtreepredesc}{\glstreepredesc}

```

reechildpredesc

```

15321 \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

```

15322 \newcommand{\glstreedesc}[1]{%
15323 \glsxtrtreepredesc\glossentrydesc{#1}\glspostdescription
15324 }

```

\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.

```

15325 \newcommand{\glstreeDescLoc}[2]{%
15326 \ifglshasdesc{#1}%
15327 {\glstreedesc{#1}\glstreeprelocation}%
15328 {\ifglshassymbol{#1}{\glstreeprelocation}{\glstreeNoDescSymbolPreLocation}}}%
15329 #2%
15330 }

```

mbolPreLocation

```
\glstreeNoDescSymbolPreLocation
```

```

15331 \newcommand{\glstreeNoDescSymbolPreLocation}{\space}

```

Similarly for the symbol.

`\glstreesymbol`

```
15332 \newcommand{\glstreesymbol}[1]{%
15333   \ifglshassymbol{#1}{\space(\glossentrysymbol{#1})}{}%
15334 }%
```

And for the child entries:

`glstreechilddesc`

```
15335 \newcommand{\glstreechilddesc}[1]{%
15336   \glxtrtreechildpredesc\glossentrydesc{#1}\glspostdescription
15337 }%
```

`glstreeChildDescLoc`

```
15338 \newcommand{\glstreeChildDescLoc}[2]{%
15339   \ifglshasdesc{#1}%
15340   {\glstreechilddesc{#1}\glstreechildprelocation}%
15341   {\ifglshassymbol{#1}{\glstreechildprelocation}%
15342     {\glstreeNoDescSymbolPreLocation}%
15343   }%
15344   #2%
15345 }%
```

`glstreechildsymbol` This just behaves in the same way as the top-level.

```
15346 \newcommand{\glstreechildsymbol}[1]{%
15347   \glstreesymbol{#1}%
15348 }%

15349 \renewglossarystyle{tree}{%
15350   \renewenvironment{theglossary}%
15351     {\setlength{\parindent}{0pt}%
15352     \setlength{\parskip}{0pt plus 0.3pt}}%
15353   {}%
15354   \renewcommand*{\glossaryheader}{}%
15355   \renewcommand*{\glsgroupheading}[1]{}%
15356   \renewcommand{\glossentry}[2]{%
15357     \hangindent0pt\relax
15358     \parindent0pt\relax
15359     \glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
15360     \glstreesymbol{##1}%
15361     \glstreeDescLoc{##1}{##2}\par
15362   }%
15363   \renewcommand{\subglossentry}[3]{%
15364     \hangindent##1\glstreeindent\relax
15365     \parindent##1\glstreeindent\relax
15366     \ifnum##1=1\relax
15367       \glssubentryitem{##2}%
15368     \fi
15369     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%

```

```

15370      \glstreechildsymbol{##2}%
15371      \glstreeChildDescLoc{##2}{##3}\par
15372    }%
15373    \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\glstreegroupskip\fi}%
15374  }%
15375 }
15376 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

15377 \ifdef{\@glsstyle@treegroup}
15378 {%
15379   \renewglossarystyle{treegroup}{%
15380     \setglossarystyle{tree}%
15381     \renewcommand{\glsgroupheading}[1]{%
15382       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
15383       \glstreePreHeader{##1}{\glstr@grptitle}%
15384       \par\noindent\glstreegroupheaderfmt{\glstr@grptitle}%
15385       \glstreegroupheaderskip\@afterheading}%
15386   }
15387 }
15388 {}

```

Similarly for treehypergroup

```

15389 \ifdef{\@glsstyle@treehypergroup}
15390 {%
15391   \renewglossarystyle{treehypergroup}{%
15392     \setglossarystyle{tree}%
15393     \renewcommand*{\glossaryheader}{%
15394       \par\noindent\glstreenavigationfmt{\glsnavigation}%
15395       \glstreegroupheaderskip\@afterheading}%
15396     \renewcommand*{\glsgroupheading}[1]{%
15397       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
15398       \glstreePreHeader{##1}{\glstr@grptitle}%
15399       \par\noindent
15400       \glstreegroupheaderfmt
15401         {\glsnavhypertarget{##1}{\glstr@grptitle}}%
15402       \glstreegroupheaderskip\@afterheading}%
15403   }
15404 }
15405 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

15406 \ifdef{\@glsstyle@treenoname}
15407 {%

```

Provide a command for use with the treenoname styles that displays the pre-description separator, the description and post-description hook.

streenonamedesc

```

15408   \newcommand{\glstreenonamedesc}[1]{%
15409     \glstreepredesc\glossentrydesc{#1}\glspostdescription

```

15410 }%

Similarly for the symbol.

treenonamesymbol

```
15411 \newcommand{\glstreenonamesymbol}[1]{%
15412   \ifglshassymbol{#1}{\space(\glossentrysymbol{#1})}{}%
15413 }%
```

treenonameDescLoc

```
15414 \newcommand{\glstreenonameDescLoc}[2]{%
15415   \glstreenonamedesc{#1}\glstreeprelocation#2%
15416 }
```

nonamechilddesc The child entry doesn't have the pre-description separator as the name isn't displayed.

```
15417 \newcommand{\glstreenonamechilddesc}[1]{%
15418   \glossentrydesc{#1}\glspostdescription
15419 }%
```

nameChildDescLoc

```
15420 \newcommand{\glstreenonameChildDescLoc}[2]{%
15421   \glstreenonamechilddesc{#1}\glstreechildprelocation#2%
15422 }

15423 \renewglossarystyle{treenoname}{%
15424   \renewenvironment{theglossary}%
15425     {\setlength{\parindent}{0pt}%
15426      \setlength{\parskip}{0pt plus 0.3pt}}%
15427   {%
15428     \renewcommand*{\glossaryheader}{}%
15429     \renewcommand*{\glsgroupheading}[1]{}%
15430     \renewcommand{\glossentry}[2]{%
15431       \hangindent0pt\relax
15432       \parindent0pt\relax
15433       \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
15434       \glstreenonamesymbol{##1}%

15435       \glstreenonameDescLoc{##1}{##2}\par
15436     }%
15437     \renewcommand{\subglossentry}[3]{%
15438       \hangindent##1\glstreeindent\relax
15439       \parindent##1\glstreeindent\relax
15440       \ifnum##1=1\relax
15441         \glssubentryitem{##2}%
15442       \fi
15443       \glstarget{##2}{\strut}%
15444       \glstreenonameChildDescLoc{##2}{##3}\par
15445     }%
15446     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\glstreegroupskip\fi}%
15447   }
```



```
15448 }
15449 {}
```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```
15450 \ifdef{\@glsstyle@treenonamegroup}
15451 {%
15452   \renewglossarystyle{treenonamegroup}{%
15453     \setglossarystyle{treenoname}%
15454     \renewcommand{\glsgroupheading}[1]{%
15455       \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
15456       \glstreePreHeader{##1}{\glsxtr@grptitle}%
15457       \par\noindent\glstreegroupheaderfmt{\glsxtr@grptitle}%
15458       \glstreegroupheaderskip\@afterheading
15459     }%
15460   }
15461 }
15462 {}
```

Similarly for `treenonamehypergroup`

```
15463 \ifdef{\@glsstyle@treenonamehypergroup}
15464 {%
15465   \renewglossarystyle{treenonamehypergroup}{%
15466     \setglossarystyle{treenoname}%
15467     \renewcommand*\glossaryheader{%
15468       \par\noindent\glstreenavigationfmt{\glsnavigation}%
15469       \glstreegroupheaderskip\@afterheading}%
15470     \renewcommand*\glsgroupheading[1]{%
15471       \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
15472       \glstreePreHeader{##1}{\glsxtr@grptitle}%
15473       \par\noindent
15474       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsxtr@grptitle}}%
15475       \glstreegroupheaderskip\@afterheading}%
15476   }
15477 }
15478 {}
```

The `alttree` style is redefined to make it easier to make minor adjustments.

```
15479 \ifdef{\@glsstyle@alttree}
15480 {%
```

Only redefine this style if it's already been defined.

`salttreepredesc`

```
15481 \newcommand{\glsalttreepredesc}{}

```

`reechildpredesc`

```
15482 \newcommand{\glsalttreechildpredesc}{\glsalttreepredesc}

```

`bolDescLocation`

`\glxtralttreeSymbolDescLocation{<label>}{<location list>}`

Layout the symbol, description and location for top-level entries.

```

15483 \newcommand{\glxtralttreeSymbolDescLocation}[2]{%
15484   {%
15485     \let\par\glxtrAltTreePar

15486     \let\glxtrtreepredesc\glsalttreepredesc
15487     \let\glxtrtreechildpredesc\glsalttreechildpredesc
15488     \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%

15489     \glstreeDescLoc{#1}{#2}\par
15490   }%
15491 }

```

`trAltTreeIndent` Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

15492 \newlength\glxtrAltTreeIndent

```

`lsxtrAltTreePar` Multi-paragraph descriptions need to keep the hanging indent.

```

15493 \newcommand{\glxtrAltTreePar}{%
15494   \@@par
15495   \glxtrAltTreeSetHangIndent
15496   \setlength{\parindent}{\dimexpr\hangindent+\glxtrAltTreeIndent}%
15497 }

```

`bolDescLocation`

`\glxtralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

15498 \newcommand{\glxtralttreeSubSymbolDescLocation}[3]{%
15499   \glxtralttreeSymbolDescLocation{#2}{#3}%
15500 }

```

`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.

```

15501 \newlength\glxtrtreetopindent

```

`sxtralttreeInit` User-level initialisation for the alttree style.

```

15502 \newcommand*{\glxtralttreeInit}{%
15503   \settowidth{\glxtrtreetopindent}{\glstreenamfmt{\glsgetwidestname\space}}%
15504   \glxtrAltTreeIndent=\parindent
15505 }

```

`\glssetwidest` The original `\glssetwidest` only uses `\def`. This uses `\gdef`.

```

15506 \newcommand*{\glssetwidest}[2][0]{%
15507   \csgdef{@glswidestname\romannumeral#1}{#2}%
15508 }

```

`\eglssetwidest` The original `\glssetwidest` only uses `\def`. This uses `\protected@csdef`.

```

15509 \newcommand*{\eglssetwidest}[2][0]{%
15510   \protected@csdef{@glswidestname\romannumeral#1}{#2}%
15511 }

```

`\xglssetwidest` Like the above but uses `\protected@csxdef`.

```

15512 \newcommand*{\xglssetwidest}[2][0]{%
15513   \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
15514 }

```

`glsupdatewidest` Only sets if new value is wider than old value.

```

15515 \newcommand*{\glsupdatewidest}[2][0]{%
15516   \ifcsundef{@glswidestname\romannumeral#1}%
15517   {\csdef{@glswidestname\romannumeral#1}{#2}}%
15518   {%
15519     \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
15520     \settowidth{\dimen@ii}{#2}%
15521     \ifdim\dimen@ii>\dimen@
15522       \csdef{@glswidestname\romannumeral#1}{#2}%
15523     \fi
15524   }%
15525 }

```

`glsupdatewidest` As above but global definition.

```

15526 \newcommand*{\glsupdatewidest}[2][0]{%
15527   \ifcsundef{@glswidestname\romannumeral#1}%
15528   {\csgdef{@glswidestname\romannumeral#1}{#2}}%
15529   {%
15530     \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
15531     \settowidth{\dimen@ii}{#2}%
15532     \ifdim\dimen@ii>\dimen@
15533       \csgdef{@glswidestname\romannumeral#1}{#2}%
15534     \fi
15535   }%
15536 }

```

`glsupdatewidest` As `\glsupdatewidest` but expands value.

```

15537 \newcommand*{\eglsupdatewidest}[2][0]{%
15538   \ifcsundef{@glswidestname\romannumeral#1}%
15539   {\protected@csdef{@glswidestname\romannumeral#1}{#2}}%
15540   {%
15541     \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
15542     \settowidth{\dimen@ii}{#2}%

```

```

15543      \ifdim\dimen@ii>\dimen@
15544      \protected@csedef{@glswidestname\romannumeral#1}{#2}%
15545      \fi
15546  }%
15547  }

```

glupdatewidest As above but global.

```

15548  \newcommand*{\xglupdatewidest}[2][0]{%
15549  \ifcsundef{@glswidestname\romannumeral#1}%
15550  {\protected@csxdef{@glswidestname\romannumeral#1}{#2}}%
15551  {%
15552  \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
15553  \settowidth{\dimen@ii}{#2}%
15554  \ifdim\dimen@ii>\dimen@
15555  \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
15556  \fi
15557  }%
15558  }

```

glswidestname Provide a user-level macro to obtain the widest top-level name.

```

15559  \newcommand*{\glswidestname}{\@glswidestname}

```

glswidestsubname Provide a user-level macro to obtain the widest sub-entry name.

```

15560  \newcommand*{\glswidestsubname}[1]{%
15561  \ifcsundef{@glswidestname\romannumeral#1}%
15562  {\@glswidestname}%
15563  {\csuse{@glswidestname\romannumeral#1}}%
15564  }

```

glswidestTopLevelName CamelCase is easier for long command names. Provide a CamelCase synonym of `\glswidestTopLevelName`.

```

15565  \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.

```

15566  \newrobustcmd*{\glswidestUsedTopLevelName}[1][\@glo@types]{%
15567  \dimen@=0pt\relax
15568  \gls@tmplen=0pt\relax
15569  \foralllglossaries[#1]{\@gls@type}%
15570  {%
15571  \forlsglentries[\@gls@type]{\@glo@label}%
15572  {%
15573  \ifglswidestUsed{\@glo@label}%
15574  {%
15575  \ifglshasparent{\@glo@label}%
15576  }%
15577  }%

```

```

15578         \settowidth{\dimen@}%
15579         {\glstreenamefmt{\glentryname{\@glo@label}}}%
15580         \ifdim\dimen@>\gls@tmplen
15581             \gls@tmplen=\dimen@
15582             \eglssetwidest{\glentryname{\@glo@label}}%
15583         \fi
15584     }%
15585 }%
15586 {}%
15587 }%
15588 }%
15589 }

```

destUsedAnyName Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

15590 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
15591     \dimen@=0pt\relax
15592     \gls@tmplen=0pt\relax
15593     \foralllglossaries[#1]{\@gls@type}%
15594     {%
15595         \forglentries[\@gls@type]{\@glo@label}%
15596         {%
15597             \ifglsused{\@glo@label}%
15598             {%
15599                 \settowidth{\dimen@}%
15600                 {\glstreenamefmt{\glentryname{\@glo@label}}}%
15601                 \ifdim\dimen@>\gls@tmplen
15602                     \gls@tmplen=\dimen@
15603                     \eglssetwidest{\glentryname{\@glo@label}}%
15604             \fi
15605         }%
15606     }%
15607 }%
15608 }%
15609 }

```

ndWidestAnyName Like the above but doesn't check if the entry has been used.

```

15610 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
15611     \dimen@=0pt\relax
15612     \gls@tmplen=0pt\relax
15613     \foralllglossaries[#1]{\@gls@type}%
15614     {%
15615         \forglentries[\@gls@type]{\@glo@label}%
15616         {%
15617             \settowidth{\dimen@}%
15618             {\glstreenamefmt{\glentryname{\@glo@label}}}%
15619             \ifdim\dimen@>\gls@tmplen
15620                 \gls@tmplen=\dimen@
15621                 \eglssetwidest{\glentryname{\@glo@label}}%

```

```

15622         \fi
15623     }%
15624 }%
15625 }

```

estUsedLevelTwo This is like \glsFindWidestUsedTopLevelName but also sets the first two sub-levels as well.

Any entry that has a great-grandparent is ignored.

```

15626 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@gls@types]{%
15627     \dimen@=0pt\relax
15628     \dimen@i=0pt\relax
15629     \dimen@ii=0pt\relax
15630     \forallglossaries[#1]{\@gls@type}%
15631     {%
15632         \forglsentries[\@gls@type]{\@glo@label}%
15633         {%
15634             \ifglsused{\@glo@label}%
15635             {%
15636                 \ifglshasparent{\@glo@label}%
15637                 {%
15638                     \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
15639                     \ifglshasparent{\@glo@parent}%
15640                     {%
15641                         \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
15642                         \ifglshasparent{\@glo@parent}%
15643                         {%
15644                             {%
15645                                 \settowidth{\gls@tmplen}%
15646                                     {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15647                                 \ifdim\gls@tmplen>\dimen@ii
15648                                     \dimen@ii=\gls@tmplen
15649                                     \eglssetwidest[2]{\glsentryname{\@glo@label}}%
15650                                 \fi
15651                             }%
15652                         }%
15653                     }%
15654                     \settowidth{\gls@tmplen}%
15655                         {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15656                     \ifdim\gls@tmplen>\dimen@i
15657                         \dimen@i=\gls@tmplen
15658                         \eglssetwidest[1]{\glsentryname{\@glo@label}}%
15659                     \fi
15660                 }%
15661             }%
15662         }%
15663         \settowidth{\gls@tmplen}%
15664             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15665         \ifdim\gls@tmplen>\dimen@
15666             \dimen@=\gls@tmplen
15667             \eglssetwidest{\glsentryname{\@glo@label}}%

```

```

15668         \fi
15669     }%
15670 }%
15671 {}%
15672 }%
15673 }%
15674 }

```

dWidestLevelTwo This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

15675 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
15676     \dimen@=0pt\relax
15677     \dimen@i=0pt\relax
15678     \dimen@ii=0pt\relax
15679     \foralllglossaries[#1]{\@gls@type}%
15680     {%
15681         \forglsentries[\@gls@type]{\@glo@label}%
15682         {%
15683             \ifglshasparent{\@glo@label}%
15684             {%
15685                 \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
15686                 \ifglshasparent{\@glo@parent}%
15687                 {%
15688                     \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
15689                     \ifglshasparent{\@glo@parent}%
15690                     {}%
15691                     {%
15692                         \settowidth{\gls@tmplen}%
15693                             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
15694                         \ifdim\gls@tmplen>\dimen@ii
15695                             \dimen@ii=\gls@tmplen
15696                             \eglssetwidest[2]{\glsentryname{\@glo@label}}%
15697                         \fi
15698                     }%
15699                 }%
15700             }%
15701             \settowidth{\gls@tmplen}%
15702                 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
15703             \ifdim\gls@tmplen>\dimen@i
15704                 \dimen@i=\gls@tmplen
15705                 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
15706             \fi
15707         }%
15708     }%
15709     {%
15710         \settowidth{\gls@tmplen}%
15711             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
15712         \ifdim\gls@tmplen>\dimen@
15713             \dimen@=\gls@tmplen
15714             \eglssetwidest{\glsentryname{\@glo@label}}%

```

```

15715         \fi
15716     }%
15717 }%
15718 }%
15719 }

```

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.

```

15720 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
15721     \dimen@=0pt\relax
15722     \gls@tmplen=0pt\relax
15723     #2=0pt\relax
15724     \forallglossaries[#1]{\@gls@type}%
15725     {%
15726         \forglsentries[\@gls@type]{\@glo@label}%
15727         {%
15728             \ifglsused{\@glo@label}%
15729             {%
15730                 \settowidth{\dimen@}%
15731                 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15732                 \ifdim\dimen@>\gls@tmplen
15733                     \gls@tmplen=\dimen@
15734                     \eglssetwidest{\glsentryname{\@glo@label}}%
15735                 \fi
15736                 \settowidth{\dimen@}%
15737                 {\glsentrysymbol{\@glo@label}}%
15738                 \ifdim\dimen@>#2\relax
15739                     #2=\dimen@
15740                 \fi
15741             }%
15742         }%
15743     }%
15744 }%
15745 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

15746 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
15747     \dimen@=0pt\relax
15748     \gls@tmplen=0pt\relax
15749     #2=0pt\relax
15750     \forallglossaries[#1]{\@gls@type}%
15751     {%
15752         \forglsentries[\@gls@type]{\@glo@label}%
15753         {%
15754             \settowidth{\dimen@}%
15755             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15756             \ifdim\dimen@>\gls@tmplen
15757                 \gls@tmplen=\dimen@
15758                 \eglssetwidest{\glsentryname{\@glo@label}}%

```



```

15759      \fi
15760      \settowidth{\dimen@}%
15761      {\glsentrysymbol{\@glo@label}}}%
15762      \ifdim\dimen@>#2\relax
15763      #2=\dimen@
15764      \fi
15765    }%
15766  }%
15767 }

```

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.

```

15768 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
15769   \dimen@=0pt\relax
15770   \gls@tmplen=0pt\relax
15771   #2=0pt\relax
15772   #3=0pt\relax
15773   \forallglossaries[#1]{\@gls@type}%
15774   {%
15775     \forglentries[\@gls@type]{\@glo@label}%
15776     {%
15777       \ifglused{\@glo@label}%
15778       {%
15779         \settowidth{\dimen@}%
15780         {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15781         \ifdim\dimen@>\gls@tmplen
15782         \gls@tmplen=\dimen@
15783         \eglssetwidest{\glsentryname{\@glo@label}}%
15784         \fi
15785         \settowidth{\dimen@}%
15786         {\glsentrysymbol{\@glo@label}}}%
15787         \ifdim\dimen@>#2\relax
15788         #2=\dimen@
15789         \fi
15790         \settowidth{\dimen@}%
15791         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
15792         \ifdim\dimen@>#3\relax
15793         #3=\dimen@
15794         \fi
15795       }%
15796     }%
15797   }%
15798 }%
15799 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

15800 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%

```

```

15801 \dimen@=0pt\relax
15802 \gls@tmplen=0pt\relax
15803 #2=0pt\relax
15804 #3=0pt\relax
15805 \forallglossaries[#1]{\@gls@type}%
15806 {%
15807   \forglsentries[\@gls@type]{\@glo@label}%
15808   {%
15809     \settowidth{\dimen@}%
15810     {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15811     \ifdim\dimen@>\gls@tmplen
15812       \gls@tmplen=\dimen@
15813       \eglssetwidest{\glsentryname{\@glo@label}}%
15814     \fi
15815     \settowidth{\dimen@}%
15816     {\glsentrysymbol{\@glo@label}}%
15817     \ifdim\dimen@>#2\relax
15818       #2=\dimen@
15819     \fi
15820     \settowidth{\dimen@}%
15821     {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
15822     \ifdim\dimen@>#3\relax
15823       #3=\dimen@
15824     \fi
15825   }%
15826 }%
15827 }

```

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.

```

15828 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
15829   \dimen@=0pt\relax
15830   \gls@tmplen=0pt\relax
15831   #2=0pt\relax
15832   \forallglossaries[#1]{\@gls@type}%
15833   {%
15834     \forglsentries[\@gls@type]{\@glo@label}%
15835     {%
15836       \ifglsused{\@glo@label}%
15837       {%
15838         \settowidth{\dimen@}%
15839         {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15840         \ifdim\dimen@>\gls@tmplen
15841           \gls@tmplen=\dimen@
15842           \eglssetwidest{\glsentryname{\@glo@label}}%
15843         \fi
15844         \settowidth{\dimen@}%
15845         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%

```

```

15846         \ifdim\dimen@>#2\relax
15847         #2=\dimen@
15848         \fi
15849     }%
15850 }%
15851 }%
15852 }%
15853 }

```

AnyNameLocation Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

15854 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
15855     \dimen@=0pt\relax
15856     \gls@tmplen=0pt\relax
15857     #2=0pt\relax
15858     \forallglossaries[#1]{\@gls@type}%
15859     {%
15860         \forglsentries[\@gls@type]{\@glo@label}%
15861         {%
15862             \settowidth{\dimen@}%
15863             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15864             \ifdim\dimen@>\gls@tmplen
15865                 \gls@tmplen=\dimen@
15866                 \eglssetwidest{\glsentryname{\@glo@label}}%
15867             \fi
15868             \settowidth{\dimen@}%
15869             {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
15870             \ifdim\dimen@>#2\relax
15871                 #2=\dimen@
15872             \fi
15873         }%
15874     }%
15875 }

```

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`.

```

15876 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
15877     \glstreeindent=\glsxtrtreetopindent\relax
15878 }

```

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.

```

15879 \newcommand*{\glxtrComputeTreeSubIndent}[3]{%
15880   \ifcsundef{@glswidestname\romannumeral#1}%
15881   {%
15882     \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
15883   }%
15884   {%
15885     \settowidth{#3}{\glstreenamefmt{%
15886       \csname @glswidestname\romannumeral#1\endcsname\space}}%
15887   }%
15888 }

```

eeSetHangIndent Set \hangindent for top-level entries:

```

15889 \newcommand*{\glxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set \hangindent for sub-entries:

```

15890 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine alttree:

```

15891 \renewglossarystyle{alttree}{%
15892   \renewenvironment{theglossary}%
15893   {%
15894     \glxtralttreeInit
15895     \def\@gls@prevlevel{-1}%
15896     \mbox{}\par}%
15897   {\par}%
15898   \renewcommand*{\glossaryheader}{}%
15899   \renewcommand*{\glsgroupheading}[1]{}%
15900   \renewcommand{\glossentry}[2]{%
15901     \ifnum\@gls@prevlevel=0\relax
15902     \else
15903       \glxtrComputeTreeIndent{##1}%
15904     \fi
15905     \parindent\glstreeindent
15906     \glxtrAltTreeSetHangIndent
15907     \makebox[0pt][r]{%
15908       {%
15909         \glstreenamebox{\glstreeindent}%
15910         {%
15911           \glsentryitem{##1}%
15912           \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
15913         }%
15914       }%
15915       \glxtralttreeSymbolDescLocation{##1}{##2}%
15916       \def\@gls@prevlevel{0}%
15917     }
15918     \renewcommand{\subglossentry}[3]{%
15919       \ifnum##1=1\relax
15920       \glssubentryitem{##2}%
15921     \fi

```

```

15922 \ifnum\@gls@prevlevel=##1\relax
15923 \else
15924 \glstrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
15925 \ifnum\@gls@prevlevel<##1\relax
15926 \setlength\glstreeindent\gls@tmplen
15927 \addtolength\glstreeindent\parindent
15928 \parindent\glstreeindent
15929 \else
15930 \ifnum\@gls@prevlevel=0\relax
15931 \glstrComputeTreeIndent{##2}%
15932 \else
15933 \glstrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
15934 \fi
15935 \addtolength\parindent{-\glstreeindent}%
15936 \setlength\glstreeindent\parindent
15937 \fi
15938 \fi
15939 \glstrAltTreeSetSubHangIndent{##1}%
15940 \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
15941 \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
15942 \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
15943 \def\@gls@prevlevel{##1}%
15944 }%
15945 \renewcommand*\@glsgroupskip{\ifglsnogroupskip\else\glstreegroupskip\fi}%
15946 }
15947 }%
15948 {%
15949 }

```

Redefine `alttreegroup` so that it discourages a break after group headings.

```

15950 \ifdef{\@glsstyle@alttreegroup}
15951 {%
15952 \renewglossarystyle{alttreegroup}{%
15953 \setglossarystyle{alttree}%
15954 \renewcommand{\glsgroupheading}[1]{\par
15955 \def\@gls@prevlevel{-1}%
15956 \hangindent0pt\relax
15957 \parindent0pt\relax
15958 \glstrgetgrouptitle{##1}{\glstr@grptitle}%
15959 \glstreePreHeader{##1}{\glstr@grptitle}%
15960 \glstreegroupheaderfmt{\glstr@grptitle}%

```

Can't use `\@afterheading` here as it messes with the first item of the group.

```

15961 \glstreegroupheaderskip
15962 }%
15963 }%
15964 }%
15965 {%
15966 }

```

Similarly for `alttreehypergroup`.

```

15967 \ifdef{\@glsstyle@almtreehypergroup}
15968 {%
15969   \renewglossarystyle{almtreehypergroup}{%
15970     \setglossarystyle{almtree}%
15971     \renewcommand*{\glossaryheader}{%
15972       \par
15973       \def\@gls@prevlevel{-1}%
15974       \hangindent0pt\relax
15975       \parindent0pt\relax
15976       \glstreenavigationfmt{\glsnavigation}%

```

Can't use \@afterheading here as it messes with the first item of the group.

```

15977   \glstreegroupheaderskip
15978 }%
15979 \renewcommand*{\glsgroupheading}[1]{%
15980   \glstrgetgrouptitle{##1}{\glstr@grptitle}%
15981   \glstreePreHeader{##1}{\glstr@grptitle}%
15982   \par
15983   \def\@gls@prevlevel{-1}%
15984   \hangindent0pt\relax
15985   \parindent0pt\relax
15986   \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glstr@grptitle}}}%

```

Can't use \@afterheading here as it messes with the first item of the group.

```

15987   \glstreegroupheaderskip
15988 }%
15989 }
15990 }%
15991 {%
15992 }

```

2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

15993 \ifdef{\@glsstyle@mcolindexgroup}
15994 {%
15995   \renewglossarystyle{mcolindexgroup}{%
15996     \setglossarystyle{mcolindex}%
15997     \renewcommand*{\glsgroupheading}[1]{%
15998       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
15999       \glstreePreHeader{##1}{\glstr@grptitle}%
16000       \item\glstreegroupheaderfmt{\glstr@grptitle}%
16001       \glstreegroupheaderskip\@afterheading
16002     }%
16003   }
16004 }%
16005 {%
16006 }

```

Similarly for mcolindexhypergroup.

```

16007 \ifdef{\@glsstyle@mcolindexhypergroup}
16008 {%
16009   \renewglossarystyle{mcolindexhypergroup}{%
16010     \setglossarystyle{mcolindex}%
16011     \renewcommand*{\glossaryheader}{%
16012       \item\glstreenavigationfmt{\glsnavigation}%

16013       \glstreegroupheaderskip\@afterheading
16014     }%
16015     \renewcommand*{\glsgroupheading}[1]{%
16016       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16017       \glstreePreHeader{##1}{\glstr@grptitle}%
16018       \item\glstreegroupheaderfmt
16019         {\glsnavhypertarget{##1}{\glstr@grptitle}}%
16020       \glstreegroupheaderskip\@afterheading
16021     }%
16022   }
16023 }%
16024 {%
16025 }

```

Similarly for mcolindexspannav.

```

16026 \ifdef{\@glsstyle@mcolindexspannav}
16027 {%
16028   \renewglossarystyle{mcolindexspannav}{%
16029     \setglossarystyle{index}%
16030     \renewenvironment{theglossary}%
16031     {%
16032       \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]{%
16033         \setlength{\parindent}{0pt}%
16034         \setlength{\parskip}{0pt plus 0.3pt}%
16035         \let\item\glstreeitem}%
16036       \end{multicols}}%
16037     \renewcommand*{\glsgroupheading}[1]{%
16038       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16039       \glstreePreHeader{##1}{\glstr@grptitle}%
16040       \item\glstreegroupheaderfmt
16041         {\glsnavhypertarget{##1}{\glstr@grptitle}}%
16042       \glstreegroupheaderskip\@afterheading
16043     }%
16044   }
16045 }%
16046 {%
16047 }

```

Similarly for mcoltreegroup.

```

16048 \ifdef{\@glsstyle@mcoltreegroup}
16049 {%
16050   \renewglossarystyle{mcoltreegroup}{%

```

```

16051 \setglossarystyle{mcoltree}%
16052 \renewcommand{\glsgroupheading}[1]{%
16053 \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16054 \glstreePreHeader{##1}{\glxtr@grptitle}%
16055 \par\noindent\glstreegroupheaderfmt{\glxtr@grptitle}%
16056 \glstreegroupheaderskip\@afterheading
16057 }%
16058 }
16059 }%
16060 {%
16061 }

```

Similarly for mcoltreehypergroup.

```

16062 \ifdef{\@glsstyle@mcoltreehypergroup}
16063 {%
16064 \renewglossarystyle{mcoltreehypergroup}{%
16065 \setglossarystyle{mcoltree}%
16066 \renewcommand*\glsgroupheading{%
16067 \par\noindent\glstreenavigationfmt{\glsnavigation}%
16068 \glstreegroupheaderskip
16069 }%
16070 \renewcommand*\glsgroupheading}[1]{%
16071 \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16072 \glstreePreHeader{##1}{\glxtr@grptitle}%
16073 \par\noindent
16074 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}%
16075 \glstreegroupheaderskip\@afterheading
16076 }%
16077 }
16078 }%
16079 {%
16080 }

```

Similarly for mcoltreespannav.

```

16081 \ifdef{\@glsstyle@mcoltreespannav}
16082 {%
16083 \renewglossarystyle{mcoltreespannav}{%
16084 \setglossarystyle{tree}%
16085 \renewenvironment{theglossary}%
16086 {%
16087 \begin{multicols}{\glsmcols}%
16088 [\noindent\glstreenavigationfmt{\glsnavigation}]]%
16089 \setlength{\parindent}{0pt}%
16090 \setlength{\parskip}{0pt plus 0.3pt}%
16091 }%
16092 {\end{multicols}}}%
16093 \renewcommand*\glsgroupheading}[1]{%
16094 \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16095 \glstreePreHeader{##1}{\glxtr@grptitle}%
16096 \par\noindent

```



```

16097      \glstreegroupheaderfmt{\glsnavigationtarget{##1}{\glstr@grptitle}}}%
16098      \glstreegroupheaderskip\@afterheading
16099  }%
16100  }
16101 }%
16102 {%
16103 }

```

Similarly for mcoltreenonamegroup.

```

16104 \ifdef{\@glsstyle@mcoltreenonamegroup}
16105 {%
16106   \renewglossarystyle{mcoltreenonamegroup}{%
16107     \setglossarystyle{mcoltreenoname}%
16108     \renewcommand{\glsgroupheading}[1]{%
16109       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16110       \glstreePreHeader{##1}{\glstr@grptitle}%
16111       \par\noindent\glstreegroupheaderfmt{\glstr@grptitle}%
16112       \glstreegroupheaderskip\@afterheading
16113     }%
16114   }
16115 }%
16116 {%
16117 }

```

Similarly for mcoltreenonamehypergroup.

```

16118 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
16119 {%
16120   \renewglossarystyle{mcoltreenonamehypergroup}{%
16121     \setglossarystyle{mcoltreenoname}%
16122     \renewcommand*\{glossaryheader}{%
16123       \par\noindent\glstreenavigationfmt{\glsnavigation}%
16124       \glstreegroupheaderskip
16125     }%
16126     \renewcommand*\{glsgroupheading}[1]{%
16127       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16128       \glstreePreHeader{##1}{\glstr@grptitle}%
16129       \par\noindent
16130       \glstreegroupheaderfmt{\glsnavigationtarget{##1}{\glstr@grptitle}}}%
16131       \glstreegroupheaderskip\@afterheading}%
16132   }
16133 }%
16134 {%
16135 }

```

Similarly for mcoltreenonamespannav.

```

16136 \ifdef{\@glsstyle@mcoltreenonamespannav}
16137 {%
16138   \renewglossarystyle{mcoltreenonamespannav}{%
16139     \setglossarystyle{treenoname}%
16140     \renewenvironment{theglossary}%
16141     {%

```

```

16142     \begin{multicols}{\glsmcols}%
16143     [\noindent\glstreenavigationfmt{\glsnavigation}]%
16144     \setlength{\parindent}{0pt}%
16145     \setlength{\parskip}{0pt plus 0.3pt}%
16146 }%
16147 {\end{multicols}}}%
16148 \renewcommand*{\glsgroupheading}[1]{%
16149   \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16150   \glstreePreHeader{##1}{\glxtr@grptitle}%
16151   \par\noindent
16152   \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}%
16153   \glstreegroupheaderskip\@afterheading}%
16154 }
16155 }%
16156 {%
16157 }

```

`mcocalttree` needs adjusting so that it uses `\glxtralttreeInit` This doesn't use `\mbox{\par` which would unbalance the top of the columns.

```

16158 \ifdef{\@glsstyle@mcocalttree}
16159 {%
16160   \renewglossarystyle{mcocalttree}{%
16161     \setglossarystyle{almtree}%
16162     \renewenvironment{theglossary}%
16163     {%
16164       \glxtralttreeInit
16165       \def\@gls@prevlevel{-1}%
16166       \begin{multicols}{\glsmcols}%
16167     }%
16168     {\par\end{multicols}}}%
16169   }
16170 }%
16171 {%
16172 }

```

Redefine `mcocalttreegroup` to discourage page breaks after the group headings.

```

16173 \ifdef{\@glsstyle@mcocalttreegroup}
16174 {%
16175   \renewglossarystyle{mcocalttreegroup}{%
16176     \setglossarystyle{mcocalttree}%
16177     \renewcommand{\glsgroupheading}[1]{%
16178       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16179       \glstreePreHeader{##1}{\glxtr@grptitle}%
16180       \par
16181       \def\@gls@prevlevel{-1}%
16182       \hangindentOpt\relax
16183       \parindentOpt\relax
16184       \glstreegroupheaderfmt{\glxtr@grptitle}%
16185       \glstreegroupheaderskip
16186     }%

```

```

16187 }
16188 }%
16189 {%
16190 }

```

Similarly for mcolalttreehypergroup.

```

16191 \ifdef{\@glsstyle@mcolalttreehypergroup}
16192 {%
16193   \renewglossarystyle{mcolalttreehypergroup}{%
16194     \setglossarystyle{mcolalttree}%
16195     \renewcommand*{\glossaryheader}{%
16196       \par
16197       \def\@gls@prevlevel{-1}%
16198       \hangindent0pt\relax
16199       \parindent0pt\relax
16200       \glstreenavigationfmt{\glsnavigation}%
16201       \glstreegroupheaderskip
16202     }%
16203     \renewcommand*{\glsgroupheading}[1]{%
16204       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16205       \glstreePreHeader{##1}{\glxtr@grptitle}%
16206       \par
16207       \def\@gls@prevlevel{-1}%
16208       \hangindent0pt\relax
16209       \parindent0pt\relax
16210       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}%
16211       \glstreegroupheaderskip
16212     }%
16213   }
16214 }%
16215 {%
16216 }

```

Similarly for mcolalttreespannav.

```

16217 \ifdef{\@glsstyle@mcolalttreespannav}
16218 {%
16219   \renewglossarystyle{mcolalttreespannav}{%
16220     \setglossarystyle{alttree}%
16221     \renewenvironment{theglossary}%
16222     {%
16223       \glxtralttreeInit
16224       \def\@gls@prevlevel{-1}%
16225       \begin{multicols}{\glsmcols}%
16226       [\noindent\glstreenavigationfmt{\glsnavigation}]%
16227     }%
16228     {\par\end{multicols}}%
16229     \renewcommand*{\glsgroupheading}[1]{%
16230       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16231       \glstreePreHeader{##1}{\glxtr@grptitle}%
16232       \par

```

```

16233     \def\@gls@prevlevel{-1}%
16234     \hangindent0pt\relax
16235     \parindent0pt\relax
16236     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\@glsxtr@grptitle}}}%
16237     \glstreegroupheaderskip
16238   }%
16239 }
16240}%
16241{%
16242}

```

Reset the default style

```

16243 \ifx\@glossary@default@style\relax
16244 \else
16245   \setglossarystyle{\@glsxtr@current@style}
16246 \fi

```

3 bookindex style (glossary-bookindex.sty)

3.1 Package Initialisation and Options

```
16247 \NeedsTeXFormat{LaTeX2e}
16248 \ProvidesPackage{glossary-bookindex}[2021/09/20 v1.46 (NLCT)]
```

Load required packages.

```
16249 \RequirePackage{multicol}
16250 \RequirePackage{glossary-tree}
```

trbookindexcols Number of columns.

```
16251 \newcommand{\glstrbookindexcols}{2}
```

trbookindexname Format used for top-level entries. (Argument is the label.)

```
16252 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

ookindexsubname Format used for sub entries.

```
16253 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

xsxtrprelocation Provide in case glossaries-stylemods isn't loaded.

```
16254 \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.

```
16255 \newcommand*{\glsxtrbookindexprelocation}[1]{%
16256    \glsxtrifhasfield{location}{#1}%
16257    {,\glsxtrprelocation}%
16258    {\glsxtrprelocation}%
16259 }
```

xsubprelocation Separator used before location list for sub-entries.

```
16260 \newcommand*{\glsxtrbookindexsubprelocation}[1]{%
16261    \glsxtrbookindexprelocation{#1}%
16262 }
```

okindexlocation

```
\glsxtrbookindexlocation{\label}{\location}
```

Displays the location.

```
16263 \newcommand*{\glsxtrbookindexlocation}[2]{#2}
```

ndexsublocation

```
\glxtrbookindexlocation{<label>}{<location>}
```

Displays the location for sub-entries.

```
16264 \newcommand*{\glxtrbookindexsublocation}{\glxtrbookindexlocation}
```

xparentchildsep Separator used between top-level parent and child entry.

```
16265 \newcommand{\glxtrbookindexparentchildsep}{\nopagebreak}
```

rentsubchildsep Separator used between sub-level parent and child entry.

```
16266 \newcommand{\glxtrbookindexparentsubchildsep}{\glxtrbookindexparentchildsep}
```

ookindexbetween Between two top-level entries identified by the labels in the arguments.

```
16267 \newcommand{\glxtrbookindexbetween}[2]{}%
```

indexsubbetween Between two level 1 entries identified by the labels in the arguments.

```
16268 \newcommand{\glxtrbookindexsubbetween}[2]{}%
```

exsubsubbetween Between two level 2 entries identified by the labels in the arguments.

```
16269 \newcommand{\glxtrbookindexsubsubbetween}[2]{}%
```

indexatendgroup At the end of a letter group. The argument is the index of the last top-level entry.

```
16270 \newcommand{\glxtrbookindexatendgroup}[1]{}%
```

exsubatendgroup At the end of a letter group. The argument is the index of the last level 1 entry.

```
16271 \newcommand{\glxtrbookindexsubatendgroup}[1]{}%
```

subsubatendgroup At the end of a letter group. The argument is the index of the last level 2 entry.

```
16272 \newcommand{\glxtrbookindexsubsubatendgroup}[1]{}%
```

kindexgroupskip Group separator.

```
16273 \newcommand{\glxtrbookindexgroupskip}{\ifglxnogroupskip\else\indexspace\fi}
```

Format group title.

dexformatheader Group separator.

```
16274 \newcommand*{\glxtrbookindexformatheader}[1]{%
16275   \par{\centering\glstreegroupheaderfmt{#1}\par}%
16276 }
```

okindexbookmark Book mark group heading if supported.

```
16277 \ifdef\pdfbookmark
16278 {%
16279   \newcommand*{\glxtrbookindexbookmark}[2]{%
16280     \ifdefstring{@@glossarysec}{chapter}%
16281     {\pdfbookmark[1]{#1}{#2}}%
16282   }
16283 }
```

```

16282     {\pdfbookmark[2]{#1}{#2}}%
16283   }
16284 }
16285 {%
16286   \newcommand*{\glxtrbookindexbookmark}[2]{%
16287 }

```

bookmarkprefix Make the bookmark label prefix used for letter groups depend on the glossary label (instead of original hardcoded “index.”).

```

16288 \newcommand*{\glxtrbookindexbookmarkprefix}{\currentglossary.}

```

indexcolspread

```

16289 \newcommand*{\glxtrbookindexcolspread}{}

```

indexmulticolseenv

```

16290 \newcommand*{\glxtrbookindexmulticolseenv}{multicols}

```

Define the style.

```

16291 \newglossarystyle{bookindex}{%
16292   \setglossarystyle{index}%
16293   \renewenvironment{theglossary}%
16294   {%
16295     \ifnum\glxtrbookindexcols>1\relax
16296       \ifdefempty\glxtrbookindexcolspread
16297       {%
16298         \edef\glxtr@beginbookindex{%
16299           \noexpand\begin{\glxtrbookindexmulticolseenv}
16300             {\glxtrbookindexcols}%
16301         }%
16302       }%
16303     {%
16304       \edef\glxtr@beginbookindex{%
16305         \noexpand\begin{\glxtrbookindexmulticolseenv}%
16306           {\glxtrbookindexcols}{\glxtrbookindexcolspread}%
16307       }%
16308     }%
16309   \else
16310     \def\glxtr@beginbookindex{%
16311   \fi
16312   \glxtr@beginbookindex
16313   \setlength{\parindent}{0pt}%
16314   \setlength{\parskip}{0pt plus 0.3pt}%
16315   \let\@glxtr@bookindex@sep\glxtrbookindexparentchildsep
16316   \let\@glxtr@bookindex@subsep\glxtrbookindexparentsubchildsep
16317   \let\@glxtr@bookindex@between\@gobble
16318   \let\@glxtr@bookindex@subbetween\@gobble
16319   \let\@glxtr@bookindex@subsubbetween\@gobble
16320   \let\@glxtr@bookindex@atendgroup\relax
16321   \let\@glxtr@bookindex@subatendgroup\relax

```

```

16322 \let\@glsxtr@bookindex@subsubatendgroup\relax
16323 \let\@glsxtr@bookindexgroupskip\relax
16324 }%
16325 {%

```

Do end group hooks.

```

16326 \@glsxtr@bookindex@subsubatendgroup
16327 \@glsxtr@bookindex@subatendgroup
16328 \@glsxtr@bookindex@atendgroup

```

End multicol environment.

```

16329 \ifnum\glsxtrbookindexcols>1\relax
16330 \edef\glsxtr@endbookindex{%
16331 \noexpand\end{\glsxtrbookindexmulticolscenv}%
16332 }%
16333 \else
16334 \def\glsxtr@endbookindex{%
16335 \fi
16336 \glsxtr@endbookindex
16337 }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

16338 \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

16339 \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

16340 \@glsxtr@bookindex@between{##1}%

```

Update separators.

```

16341 \let\@glsxtr@bookindex@sep\glsxtrbookindexparentchildsep
16342 \let\@glsxtr@bookindex@subsep\glsxtrbookindexparentsubchildsep
16343 \let\@glsxtr@bookindex@subbetween\@gobble
16344 \let\@glsxtr@bookindex@subsubbetween\@gobble
16345 \edef\@glsxtr@bookindex@between{%
16346 \noexpand\glsxtrbookindexbetween{##1}%
16347 }%
16348 \edef\@glsxtr@bookindex@atendgroup{%
16349 \noexpand\glsxtrbookindexatendgroup{##1}%
16350 }%
16351 \let\@glsxtr@bookindex@subatendgroup\relax
16352 \let\@glsxtr@bookindex@subsubatendgroup\relax

```

Format entry.

```

16353 \glstreeitem
16354 \glsentryitem{##1}%
16355 \glstarget{##1}{\glsxtrbookindexname{##1}}%
16356 \glsxtrbookindexprelocation{##1}%
16357 \glsxtrbookindexlocation{##1}{##2}%
16358 }%
16359 \renewcommand{\subglossentry}[3]{%
16360 \ifcase##1\relax

```


Level 0 (shouldn't happen as that's formatted with \glossentry).

```
16361      \glstreeitem
16362      \or
```

Level 1.

```
16363      \@glxtr@bookindex@sep
16364      \@glxtr@bookindex@subbetween{##2}%
16365      \let\@glxtr@bookindex@sep\relax
```

Update separators.

```
16366      \let\@glxtr@bookindex@subsubbetween\@gobble
16367      \let\@glxtr@bookindex@subsep\glxtrbookindexparentsubchildsep
16368      \edef\@glxtr@bookindex@subbetween{%
16369          \noexpand\glxtrbookindexsubbetween{##2}%
16370      }%
16371      \edef\@glxtr@bookindex@atsubendgroup{%
16372          \noexpand\glxtrbookindexatsubendgroup{##1}%
16373      }%
```

Start sub-item.

```
16374      \glstreesubitem
16375      \glssubentryitem{##2}%
16376      \else
```

All other levels.

```
16377      \@glxtr@bookindex@subsep
16378      \@glxtr@bookindex@subsubbetween{##2}%
```

Update separators.

```
16379      \let\@glxtr@bookindex@subsep\relax
16380      \edef\@glxtr@bookindex@subsubbetween{%
16381          \noexpand\glxtrbookindexsubsubbetween{##2}%
16382      }%
16383      \edef\@glxtr@bookindex@atsubsubendgroup{%
16384          \noexpand\glxtrbookindexatsubsubendgroup{##1}%
16385      }%
```

Start sub-sub-item.

```
16386      \glstreesubsubitem
16387      \fi
```

Format entry.

```
16388      \glstarget{##2}{\glxtrbookindexsubname{##2}}%
16389      \glxtrbookindexsubprelocation{##2}%
16390      \glxtrbookindexsublocation{##2}{##3}%
16391      }%
```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
16392      \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
16393      \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
16394 \glxtr@bookindex@subsubatendgroup
16395 \glxtr@bookindex@subatendgroup
16396 \glxtr@bookindex@atendgroup
16397 \glxtr@bookindexgroupskip
```

Update separators.

```
16398 \let\glxtr@bookindexgroupskip\glxtrbookindexgroupskip
16399 \let\glxtr@bookindex@between\@gobble
16400 \let\glxtr@bookindex@atendgroup\relax
16401 \let\glxtr@bookindex@subatendgroup\relax
16402 \let\glxtr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
16403 \glxtrgetgrouptitle{##1}{\glxtrcurrentgrptitle}%
```

Do the PDF bookmark if supported.

```
16404 \glxtrbookindexbookmark{\glxtrcurrentgrptitle}{\glxtrbookindexbookmarkprefix##1}%
```

Format the group title.

```
16405 \glxtrbookindexformatheader{\glxtrcurrentgrptitle}%
16406 \nopagebreak\indexspace\nopagebreak\@afterheading
16407 }%
16408 }
```

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.

```
16409 \newcommand{\glxtrbookindexthepage}{%
16410 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
16411 }
```

`bookindexmarkentry` Writes entry information to the `.aux` file. The argument is the entry label.

```
16412 \newcommand*{\glxtrbookindexmarkentry}[1]{%
16413 \protected@write\@auxout
16414 {\let\glxtrbookindexthepage\relax}%
16415 {\string\glxtr@setbookindexmark{\glxtrbookindexthepage}{#1}}%
16416 }
```

`setbookindexmark`

```
16417 \newcommand*{\glxtr@setbookindexmark}[2]{%
16418 \ifcsundef\glxtr@idxfirstmark@#1}%
16419 {\csgdef\glxtr@idxfirstmark@#1}{#2}}%
16420 {}%
16421 \csgdef\glxtr@idxlastmark@#1}{#2}%
16422 }
```

dexfirstmarkfmt

```
16423 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
16424   \glstryname{#1}%
16425 }
```

kindexfirstmark

```
16426 \newcommand*{\glxtrbookindexfirstmark}{%
16427   \letcs{\glxtr@label}{\glxtr@idxfirstmark@\glxtrbookindexthepage}%
16428   \ifdef\glxtr@label
16429     {\glxtrbookindexfirstmarkfmt{\glxtr@label}}%
16430   {}%
16431 }
```

ndexlastmarkfmt

```
16432 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
16433   \glstryname{#1}%
16434 }
```

okindexlastmark

```
16435 \newcommand*{\glxtrbookindexlastmark}{%
16436   \letcs{\glxtr@label}{\glxtr@idxlastmark@\glxtrbookindexthepage}%
16437   \ifdef\glxtr@label
16438     {\glxtrbookindexlastmarkfmt{\glxtr@label}}%
16439   {}%
16440 }
```

4 longextra styles (glossary-longextra.sty)

4.1 Package Initialisation and Options

Provides additional long styles.

```
16441 \NeedsTeXFormat{LaTeX2e}  
16442 \ProvidesPackage{glossary-longextra}[2021/09/20 v1.46 (NLCT)]
```

Load required packages.

```
16443 \RequirePackage{glossary-longbooktabs}
```

ongextraNameFmt

```
\glslongextraNameFmt{<label>}
```

Governs the way the name is displayed.

```
16444 \newcommand{\glslongextraNameFmt}[1]{%  
16445   \glsentryitem{#1}\glstarget{#1}{\glossentryname{#1}}%  
16446 }
```

ongextraDescFmt

```
\glslongextraDescFmt{<label>}
```

Governs the way the description is displayed.

```
16447 \newcommand{\glslongextraDescFmt}[1]{%  
16448   \glossentrydesc{#1}\glspostdescription  
16449 }
```

gextraSymbolFmt

```
\glslongextraSymbolFmt{<label>}
```

Governs the way the symbol is displayed.

```
16450 \newcommand{\glslongextraSymbolFmt}[1]{\glossentrysymbol{#1}}
```

extraLocationFmt

```
\glslongextraLocationFmt{<label>}{<location list>}
```

Governs the way the location is displayed.

```
16451 \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.

```
16452 \newcommand{\glslongextraSubNameFmt}[2]{%  
16453   \glssubentryitem{#2}\glstarget{#2}{\strut}%  
16454 }
```

extraSubDescFmt

```
\glslongextraSubDescFmt{<level>}{<label>}
```

Governs the way the child description is displayed.

```
16455 \newcommand{\glslongextraSubDescFmt}[2]{%  
16456   \glslongextraDescFmt{#2}%  
16457 }
```

extraSubSymbolFmt

```
\glslongextraSubSymbolFmt{<level>}{<label>}
```

Governs the way the child symbol is displayed.

```
16458 \newcommand{\glslongextraSubSymbolFmt}[2]{%  
16459   \glslongextraSymbolFmt{#2}%  
16460 }
```

extraSubLocationFmt

```
\glslongextraSubLocationFmt{<level>}{<label>}{<location list>}
```

Governs the way the child location list is displayed.

```
16461 \newcommand{\glslongextraSubLocationFmt}[3]{#3}
```

`gextraNameAlign` Alignment for the name column.

```
16462 \newcommand{\glslongextraNameAlign}{l}
```

`gextraDescAlign` Alignment for the description column.

```
16463 \newcommand{\glslongextraDescAlign}{>{\raggedright}p{\glsdescwidth}}
```

`extraSymbolAlign` Alignment for the symbol column.

```
16464 \newcommand{\glslongextraSymbolAlign}{c}
```

`raLocationAlign` Alignment for the location column.

```
16465 \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).

```
16466 \newcommand{\glslongextraGroupHeading}[2]{}
```

`traHeaderFormat` Format for the column headers.

```
16467 \newcommand{\glslongextraHeaderFmt}[1]{\textbf{#1}}
```

`aNameDescHeader`

```
16468 \newcommand{\glslongextraNameDescHeader}{%
16469 \glslongextraNameDescTabularHeader\endhead
16470 \glslongextraNameDescTabularFooter\endfoot
16471 }
```

`scTabularHeader`

```
16472 \newcommand{\glslongextraNameDescTabularHeader}{%
16473 \toprule
16474 \glslongextraHeaderFmt\entryname &
16475 \glslongextraHeaderFmt\descriptionname\tabularnewline
16476 \midrule
16477 }
```

`scTabularFooter`

```
16478 \newcommand{\glslongextraNameDescTabularFooter}{%
16479 \bottomrule
16480 }
```

Unlike the `alttree` style, there aren't different widths for the hierarchical levels.

`gextraSetWidest` Provide in case the tree styles haven't been loaded.

```
16481 \newcommand*{\glslongextraSetWidest}[1]{%
16482 \def\@glslongextrawidestname{#1}%
16483 }
```

`extrawidestname` Pick up the widest name from the `alttree` style if it has been set. (Will expand to nothing otherwise.)

```
16484 \newcommand*{\@glslongextrawidestname}{\csuse{@glswidestname}}
```

traUpdateWidest

```

16485 \newcommand*{\glslongextraUpdateWidest}[1]{%
16486   \ifundef\@glslongextrawidestname
16487   {\def\@glslongextrawidestname{#1}}%
16488   {%
16489     \settowidth{\dimen@}{\@glslongextrawidestname}%
16490     \settowidth{\dimen@ii}{#1}%
16491     \ifdim\dimen@ii>\dimen@
16492     \def\@glslongextrawidestname{#1}%
16493     \fi
16494   }%
16495 }
```

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`.

```

16496 \newcommand*{\glslongextraUpdateWidestChild}[2]{}
```

traSetDescWidth Computes the value of `\glsdescwidth` for the styles that only have name and description columns.

```

16497 \newcommand{\glslongextraSetDescWidth}{%
16498   \settowidth{\gls@tmplen}{\glslongextraHeaderFmt\entryname}%
16499   \settowidth{\dimen@}{\glsnamefont{\@glslongextrawidestname}}%
16500   \ifdim\dimen@>\gls@tmplen
16501     \gls@tmplen=\dimen@
16502   \fi
```

Has the widest name been set.

```

16503   \setlength{\glsdescwidth}{\dimexpr\linewidth-4\tabcolsep-\gls@tmplen}%
16504 }
```

SymSetDescWidth Computes the value of `\glsdescwidth` for the styles that only have name, symbol and description columns.

```

16505 \newcommand{\glslongextraSymSetDescWidth}{%
```

Work out the size for just the name and description style.

```

16506   \glslongextraSetDescWidth
```

Now work out the symbol column width. This is assuming that the column title will be the widest text in the column.

```

16507   \settowidth{\gls@tmplen}{\glslongextraHeaderFmt\symbolname}%
```

Subtract 2\tabcolsep and the symbol header width.

```

16508 \setlength{\glsdescwidth}{\dimexpr\glsdescwidth-2\tabcolsep-\gls@tmplen}%
16509 }

```

LocSetDescWidth Computes the value of \glsdescwidth for the styles that only have name, location and description columns.

```

16510 \newcommand{\glslongextraLocSetDescWidth}{%
  Work out the size for just the name and description style.
16511 \glslongextraSetDescWidth
  Subtract 2\tabcolsep and the location list column width.
16512 \setlength{\glsdescwidth}{\dimexpr\glsdescwidth-2\tabcolsep-\glspagelistwidth}%
16513 }

```

LocSetDescWidth Computes the value of \glsdescwidth for the styles that have name, symbol, location and description columns.

```

16514 \newcommand{\glslongextraSymLocSetDescWidth}{%
  Work out the size for just the name, symbol and description style.
16515 \glslongextraSymSetDescWidth
  Subtract 2\tabcolsep and the location list column width.
16516 \setlength{\glsdescwidth}{\dimexpr\glsdescwidth-2\tabcolsep-\glspagelistwidth}%
16517 }

```

ExtraUseTabular If true use tabular instead of longtable. Obviously only intended for short glossaries that can fit into a single page.

```

16518 \newif\ifGlsLongExtraUseTabular
16519 \GlsLongExtraUseTabularfalse

```

raTabularVAlign Only used with the tabular setting.

```

16520 \newcommand*{\glslongextraTabularVAlign}{c}

```

long-name-desc Two column style with multi-lined descriptions and header. This is similar to the longragged-booktabs style.

```

16521 \newglossarystyle{long-name-desc}%
16522 {%
16523 \ifGlsLongExtraUseTabular
16524 \renewenvironment{theglossary}%
16525 {%
16526 \glslongextraSetDescWidth
16527 \edef\@glslongextra@begintab{%
16528 \noexpand\begin{tabular}{\glslongextraTabularVAlign}{%
16529 \expandonce\glslongextraNameAlign
16530 \expandonce\glslongextraDescAlign}}%
16531 \@glslongextra@begintab
16532 }%
16533 {%

```



```

16534     \glslongextraNameDescTabularFooter
16535     \end{tabular}%
16536 }%
16537 \renewcommand*{\glossaryheader}{\glslongextraNameDescTabularHeader}%
16538 \else
16539 \renewenvironment{theglossary}%
16540 {%
16541     \glspatchLToutput
16542     \glslongextraSetDescWidth
16543     \edef\@glslongextra@begintab{%
16544         \noexpand\begin{longtable}{%
16545             \expandonce\glslongextraNameAlign
16546             \expandonce\glslongextraDescAlign}}%
16547     \@glslongextra@begintab
16548 }%
16549 {\end{longtable}}%
16550 \renewcommand*{\glossaryheader}{\glslongextraNameDescHeader}%
16551 \fi
16552 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{2}{##1}}%
16553 \renewcommand{\glossentry}[2]{%
16554     \glslongextraNameFmt{##1} &
16555     \glslongextraDescFmt{##1}\tabularnewline
16556 }%
16557 \renewcommand{\subglossentry}[3]{%
16558     \glslongextraSubNameFmt{##1}{##2}
16559     &
16560     \glslongextraSubDescFmt{##1}{##2}%
16561     \tabularnewline
16562 }%
16563 \ifglsgroupskip
16564     \renewcommand*{\glsgroupskip}{}%
16565 \else
16566     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16567 \fi
16568 }

```

cLocationHeader

```

16569 \newcommand{\glslongextraNameDescLocationHeader}{%
16570     \glslongextraNameDescLocationTabularHeader\endhead
16571     \glslongextraNameDescLocationTabularFooter\endfoot
16572 }

```

onTabularHeader

```

16573 \newcommand{\glslongextraNameDescLocationTabularHeader}{%
16574     \toprule
16575     \glslongextraHeaderFmt\entryname &
16576     \glslongextraHeaderFmt\descriptionname &
16577     \glslongextraHeaderFmt\pagelistname\tabularnewline
16578     \midrule

```

16579 }

onTabularFooter

```
16580 \newcommand{\glslongextraNameDescLocationTabularFooter}{%
16581   \bottomrule
16582 }
```

g-name-desc-loc Three columns: name, description and location list.

```
16583 \newglossarystyle{long-name-desc-loc}%
16584 {%
16585   \ifGlsLongExtraUseTabular
16586     \renewenvironment{theglossary}%
16587       {%
16588         \glslongextraLocSetDescWidth
16589         \edef\@glslongextra@begintab{%
16590           \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16591             \expandonce\glslongextraNameAlign
16592             \expandonce\glslongextraDescAlign
16593             \expandonce\glslongextraLocationAlign
16594           }}%
16595         \@glslongextra@begintab
16596       }%
16597       {%
16598         \glslongextraNameDescLocationTabularFooter
16599         \end{tabular}%
16600       }%
16601     \renewcommand*\glossaryheader{\glslongextraNameDescLocationTabularHeader}%
16602   \else
16603     \renewenvironment{theglossary}%
16604       {%
16605         \glspatchLToutput
16606         \glslongextraLocSetDescWidth
16607         \edef\@glslongextra@begintab{%
16608           \noexpand\begin{longtable}%
16609             \expandonce\glslongextraNameAlign
16610             \expandonce\glslongextraDescAlign
16611             \expandonce\glslongextraLocationAlign
16612           }}%
16613         \@glslongextra@begintab
16614       }%
16615       {\end{longtable}}%
16616     \renewcommand*\glossaryheader{\glslongextraNameDescLocationHeader}%
16617   \fi
16618   \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{3}{##1}}%
16619   \renewcommand{\glossentry}[2]{%
16620     \glslongextraNameFmt{##1} &
16621     \glslongextraDescFmt{##1} &
16622     \glslongextraLocationFmt{##1}{##2}\tabularnewline
16623   }%
```

```

16624 \renewcommand{\subglossentry}[3]{%
16625     \glslongextraSubNameFmt{##1}{##2}&
16626     \glslongextraSubDescFmt{##1}{##2}&
16627     \glslongextraSubLocationFmt{##1}{##2}{##3}%
16628     \tabularnewline
16629 }%
16630 \ifglsgroupskip
16631 \renewcommand*{\glsgroupskip}{}%
16632 \else
16633 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16634 \fi
16635 }

```

aDescNameHeader

```

16636 \newcommand{\glslongextraDescNameHeader}{%
16637 \glslongextraDescNameTabularHeader\endhead
16638 \glslongextraDescNameTabularFooter\endfoot
16639 }

```

meTabularHeader

```

16640 \newcommand{\glslongextraDescNameTabularHeader}{%
16641 \toprule
16642 \glslongextraHeaderFmt\descriptionname&
16643 \glslongextraHeaderFmt\entryname \tabularnewline
16644 \midrule
16645 }

```

meTabularFooter

```

16646 \newcommand{\glslongextraDescNameTabularFooter}{%
16647 \bottomrule
16648 }

```

long-desc-name Like name-desc but swaps the columns.

```

16649 \newglossarystyle{long-desc-name}%
16650 {%
16651 \ifGlsLongExtraUseTabular
16652 \renewenvironment{theglossary}%
16653 {%
16654     \glslongextraSetDescWidth
16655     \edef\@glslongextra@begintab{%
16656         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16657             \expandonce\glslongextraDescAlign
16658             \expandonce\glslongextraNameAlign}}%
16659     \@glslongextra@begintab
16660 }%
16661 {%
16662     \glslongextraDescNameTabularFooter
16663     \end{tabular}%
16664 }%

```

```

16665 \renewcommand*{\glossaryheader}{\glslongextraDescNameTabularHeader}%
16666 \else
16667 \renewenvironment{theglossary}%
16668 {%
16669 \glspatchLToutput
16670 \glslongextraSetDescWidth
16671 \edef\@glslongextra@begintab{%
16672 \noexpand\begin{longtable}{%
16673 \expandonce\glslongextraDescAlign
16674 \expandonce\glslongextraNameAlign}}%
16675 \@glslongextra@begintab
16676 }%
16677 {\end{longtable}}%
16678 \renewcommand*{\glossaryheader}{\glslongextraDescNameHeader}%
16679 \fi
16680 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{2}{##1}}%
16681 \renewcommand{\glossentry}[2]{%
16682 \glslongextraDescFmt{##1} &
16683 \glslongextraNameFmt{##1}\tabularnewline
16684 }%
16685 \renewcommand{\subglossentry}[3]{%
16686 \glslongextraSubDescFmt{##1}{##2} &
16687 \glslongextraSubNameFmt{##1}{##2}\tabularnewline
16688 }%
16689 \ifglsgroupskip
16690 \renewcommand*{\glsgroupskip}{}%
16691 \else
16692 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16693 \fi
16694 }

```

nDescNameHeader

```

16695 \newcommand{\glslongextraLocationDescNameHeader}{%
16696 \glslongextraLocationDescNameTabularHeader\endhead
16697 \glslongextraLocationDescNameTabularFooter\endfoot
16698 }

```

meTabularHeader

```

16699 \newcommand{\glslongextraLocationDescNameTabularHeader}{%
16700 \toprule
16701 \glslongextraHeaderFmt\pagelistname&
16702 \glslongextraHeaderFmt\descriptionname&
16703 \glslongextraHeaderFmt\entryname \tabularnewline
16704 \midrule
16705 }

```

meTabularFooter

```

16706 \newcommand{\glslongextraLocationDescNameTabularFooter}{%
16707 \bottomrule

```

16708 }

g-loc-desc-name Three columns: location, description and name.

```
16709 \newglossarystyle{long-loc-desc-name}%
16710 {%
16711   \ifGlsLongExtraUseTabular
16712     {%
16713       \glslongextraLocSetDescWidth
16714       \edef\@glslongextra@begintab{%
16715         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16716           \expandonce\glslongextraLocationAlign
16717           \expandonce\glslongextraDescAlign
16718           \expandonce\glslongextraNameAlign}}%
16719       \@glslongextra@begintab
16720     }%
16721     {%
16722       \glslongextraLocationDescNameTabularFooter
16723       \end{tabular}%
16724     }%
16725     \renewcommand*\glossaryheader{\glslongextraLocationDescNameTabularHeader}%
16726   \else
16727     \renewenvironment{theglossary}%
16728     {%
16729       \glspatchLToutput
16730       \glslongextraLocSetDescWidth
16731       \edef\@glslongextra@begintab{%
16732         \noexpand\begin{longtable}{%
16733           \expandonce\glslongextraLocationAlign
16734           \expandonce\glslongextraDescAlign
16735           \expandonce\glslongextraNameAlign}}%
16736       \@glslongextra@begintab
16737     }%
16738     {\end{longtable}}%
16739     \renewcommand*\glossaryheader{\glslongextraLocationDescNameHeader}%
16740   \fi
16741   \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{3}{##1}}%
16742   \renewcommand{\glossentry}[2]{%
16743     \glslongextraLocationFmt{##1}{##2} &
16744     \glslongextraDescFmt{##1} &
16745     \glslongextraNameFmt{##1}\tabularnewline
16746   }%
16747   \renewcommand{\subglossentry}[3]{%
16748     \glslongextraSubLocationFmt{##1}{##2}{##3} &
16749     \glslongextraSubDescFmt{##1}{##2} &
16750     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
16751   }%
16752   \ifglsnogroupskip
16753     \renewcommand*\glsgroupskip{}%
16754   \else
```

```

16755 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16756 \fi
16757 }

```

meDescSymHeader

```

16758 \newcommand{\glslongextraNameDescSymHeader}{%
16759 \glslongextraNameDescSymTabularHeader\endhead
16760 \glslongextraNameDescSymTabularFooter\endfoot
16761 }

```

ymTabularHeader

```

16762 \newcommand{\glslongextraNameDescSymTabularHeader}{%
16763 \toprule
16764 \glslongextraHeaderFmt\entryname &
16765 \glslongextraHeaderFmt\descriptionname &
16766 \glslongextraHeaderFmt\symbolname\tabularnewline
16767 \midrule
16768 }

```

ymTabularFooter

```

16769 \newcommand{\glslongextraNameDescSymTabularFooter}{%
16770 \bottomrule
16771 }

```

g-name-desc-sym Three column style with symbol in the third column.

```

16772 \newglossarystyle{long-name-desc-sym}%
16773 {%
16774 \ifGlsLongExtraUseTabular
16775 \renewenvironment{theglossary}%
16776 {%
16777 \glslongextraSymSetDescWidth
16778 \edef\@glslongextra@begintab{%
16779 \noexpand\begin{tabular}{\glslongextraTabularVAlign}{%
16780 \expandonce\glslongextraNameAlign
16781 \expandonce\glslongextraDescAlign
16782 \expandonce\glslongextraSymbolAlign
16783 }}%
16784 \@glslongextra@begintab
16785 }%
16786 {%
16787 \glslongextraNameDescSymTabularFooter
16788 \end{tabular}%
16789 }%
16790 \renewcommand*{\glossaryheader}{\glslongextraNameDescSymTabularHeader}%
16791 \else
16792 \renewenvironment{theglossary}%
16793 {%
16794 \glspatchLToutput
16795 \glslongextraSymSetDescWidth

```

```

16796 \edef\@glslongextra@begintab{%
16797 \noexpand\begin{longtable}{%
16798 \expandonce\glslongextraNameAlign
16799 \expandonce\glslongextraDescAlign
16800 \expandonce\glslongextraSymbolAlign
16801 }}%
16802 \@glslongextra@begintab
16803 }%
16804 {\end{longtable}}}%
16805 \renewcommand*{\glossaryheader}{\glslongextraNameDescSymHeader}%
16806 \fi
16807 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{3}{##1}}%
16808 \renewcommand{\glossentry}[2]{%
16809 \glslongextraNameFmt{##1} &
16810 \glslongextraDescFmt{##1} &
16811 \glslongextraSymbolFmt{##1}\tabularnewline
16812 }%
16813 \renewcommand{\subglossentry}[3]{%
16814 \glslongextraSubNameFmt{##1}{##2} &
16815 \glslongextraSubDescFmt{##1}{##2} &
16816 \glslongextraSubSymbolFmt{##1}{##2}%
16817 \tabularnewline
16818 }%
16819 \ifglsnogroupskip
16820 \renewcommand*{\glsgroupskip}{}%
16821 \else
16822 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16823 \fi
16824 }

```

mLocationHeader

```

16825 \newcommand{\glslongextraNameDescSymLocationHeader}{%
16826 \glslongextraNameDescSymLocationTabularHeader\endhead
16827 \glslongextraNameDescSymLocationTabularFooter\endfoot
16828 }

```

onTabularHeader

```

16829 \newcommand{\glslongextraNameDescSymLocationTabularHeader}{%
16830 \toprule
16831 \glslongextraHeaderFmt\entryname &
16832 \glslongextraHeaderFmt\descriptionname &
16833 \glslongextraHeaderFmt\symbolname &
16834 \glslongextraHeaderFmt\pagelistname\tabularnewline
16835 \midrule
16836 }

```

onTabularFooter

```

16837 \newcommand{\glslongextraNameDescSymLocationTabularFooter}{%
16838 \bottomrule

```

16839 }

me-desc-sym-loc Four columns: name, description and location

```
16840 \newglossarystyle{long-name-desc-sym-loc}%
16841 {%
16842   \ifGlsLongExtraUseTabular
16843     \renewenvironment{theglossary}%
16844       {%
16845         \glslongextraSymLocSetDescWidth
16846         \edef\@glslongextra@begintab{%
16847           \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16848             \expandonce\glslongextraNameAlign
16849             \expandonce\glslongextraDescAlign
16850             \expandonce\glslongextraSymbolAlign
16851             \expandonce\glslongextraLocationAlign
16852           }}%
16853         \@glslongextra@begintab
16854       }%
16855       {%
16856         \glslongextraNameDescSymLocationTabularFooter
16857         \end{tabular}%
16858       }%
16859     \renewcommand*\glossaryheader{\glslongextraNameDescSymLocationTabularHeader}%
16860   \else
16861     \renewenvironment{theglossary}%
16862       {%
16863         \glspatchLToutput
16864         \glslongextraSymLocSetDescWidth
16865         \edef\@glslongextra@begintab{%
16866           \noexpand\begin{longtable}%
16867             \expandonce\glslongextraNameAlign
16868             \expandonce\glslongextraDescAlign
16869             \expandonce\glslongextraSymbolAlign
16870             \expandonce\glslongextraLocationAlign
16871           }}%
16872         \@glslongextra@begintab
16873       }%
16874       {\end{longtable}}%
16875     \renewcommand*\glossaryheader{\glslongextraNameDescSymLocationHeader}%
16876   \fi
16877   \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{4}{##1}}%
16878   \renewcommand{\glossentry}[2]{%
16879     \glslongextraNameFmt{##1} &
16880     \glslongextraDescFmt{##1} &
16881     \glslongextraSymbolFmt{##1}&
16882     \glslongextraLocationFmt{##1}{##2}\tabularnewline
16883   }%
16884   \renewcommand{\subglossentry}[3]{%
16885     \glslongextraSubNameFmt{##1}{##2} &
```



```

16886 \glslongextraSubDescFmt{##1}{##2} &
16887 \glslongextraSubSymbolFmt{##1}{##2}&
16888 \glslongextraSubLocationFmt{##1}{##2}{##3}%
16889 \tabularnewline
16890 }%
16891 \ifglsgroupskip
16892 \renewcommand*{\glsgroupskip}{}%
16893 \else
16894 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16895 \fi
16896 }

```

meSymDescHeader

```

16897 \newcommand{\glslongextraNameSymDescHeader}{%
16898 \glslongextraNameSymDescTabularHeader\endhead
16899 \glslongextraNameSymDescTabularFooter\endfoot
16900 }

```

scTabularHeader

```

16901 \newcommand{\glslongextraNameSymDescTabularHeader}{%
16902 \toprule
16903 \glslongextraHeaderFmt\entryname &
16904 \glslongextraHeaderFmt\symbolname &
16905 \glslongextraHeaderFmt\descriptionname\tabularnewline
16906 \midrule
16907 }

```

scTabularFooter

```

16908 \newcommand{\glslongextraNameSymDescTabularFooter}{%
16909 \bottomrule
16910 }

```

g-name-sym-desc Three column style with symbol in the second column.

```

16911 \newglossarystyle{long-name-sym-desc}%
16912 {%
16913 \ifGlsLongExtraUseTabular
16914 \renewenvironment{theglossary}%
16915 {%
16916 \glslongextraSymSetDescWidth
16917 \edef\@glslongextra@begintab{%
16918 \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16919 \expandonce\glslongextraNameAlign
16920 \expandonce\glslongextraSymbolAlign
16921 \expandonce\glslongextraDescAlign
16922 }}%
16923 \@glslongextra@begintab
16924 }%
16925 {%
16926 \glslongextraNameSymDescTabularFooter

```

```

16927     \end{tabular}%
16928   }%
16929   \renewcommand*{\glossaryheader}{\glslongextraNameSymDescTabularHeader}%
16930 \else
16931   \renewenvironment{theglossary}%
16932   {%
16933     \glspatchLToutput
16934     \glslongextraSymSetDescWidth
16935     \edef\@glslongextra@begintab{%
16936       \noexpand\begin{longtable}{%
16937         \expandonce\glslongextraNameAlign
16938         \expandonce\glslongextraSymbolAlign
16939         \expandonce\glslongextraDescAlign
16940       }}%
16941     \@glslongextra@begintab
16942   }%
16943   {\end{longtable}}%
16944   \renewcommand*{\glossaryheader}{\glslongextraNameSymDescHeader}%
16945 \fi
16946 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{3}{##1}}%
16947 \renewcommand{\glossentry}[2]{%
16948   \glslongextraNameFmt{##1} &
16949   \glslongextraSymbolFmt{##1} &
16950   \glslongextraDescFmt{##1}\tabularnewline
16951 }%
16952 \renewcommand{\subglossentry}[3]{%
16953   \glslongextraSubNameFmt{##1}{##2} &
16954   \glslongextraSubSymbolFmt{##1}{##2} &
16955   \glslongextraSubDescFmt{##1}{##2}\tabularnewline
16956 }%
16957 \ifglsgroupskip
16958   \renewcommand*{\glsgroupskip}{}%
16959 \else
16960   \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16961 \fi
16962 }

```

cLocationHeader

```

16963 \newcommand{\glslongextraNameSymDescLocationHeader}{%
16964   \glslongextraNameSymDescLocationTabularHeader\endhead
16965   \glslongextraNameSymDescLocationTabularFooter\endfoot
16966 }

```

onTabularHeader

```

16967 \newcommand{\glslongextraNameSymDescLocationTabularHeader}{%
16968   \toprule
16969   \glslongextraHeaderFmt\entryname &
16970   \glslongextraHeaderFmt\symbolname &
16971   \glslongextraHeaderFmt\descriptionname &

```

```

16972 \glslongextraHeaderFmt\pagelistname\tabularnewline
16973 \midrule
16974 }

```

onTabularFooter

```

16975 \newcommand{\glslongextraNameSymDescLocationTabularFooter}{%
16976 \bottomrule
16977 }

```

me-sym-desc-loc Four column style with symbol in the second column.

```

16978 \newglossarystyle{long-name-sym-desc-loc}%
16979 {%
16980 \ifGlsLongExtraUseTabular
16981 \renewenvironment{theglossary}%
16982 {%
16983 \glslongextraSymLocSetDescWidth
16984 \edef\@glslongextra@begintab{%
16985 \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16986 \expandonce\glslongextraNameAlign
16987 \expandonce\glslongextraSymbolAlign
16988 \expandonce\glslongextraDescAlign
16989 \expandonce\glslongextraLocationAlign
16990 }}%
16991 \@glslongextra@begintab
16992 }%
16993 {%
16994 \glslongextraNameSymDescLocationTabularFooter
16995 \end{tabular}}%
16996 }%
16997 \renewcommand*\{glossaryheader}{\glslongextraNameSymDescLocationTabularHeader}%
16998 \else
16999 \renewenvironment{theglossary}%
17000 {%
17001 \glspatchLToutput
17002 \glslongextraSymLocSetDescWidth
17003 \edef\@glslongextra@begintab{%
17004 \noexpand\begin{longtable}{%
17005 \expandonce\glslongextraNameAlign
17006 \expandonce\glslongextraSymbolAlign
17007 \expandonce\glslongextraDescAlign
17008 \expandonce\glslongextraLocationAlign
17009 }}%
17010 \@glslongextra@begintab
17011 }%
17012 {\end{longtable}}%
17013 \renewcommand*\{glossaryheader}{\glslongextraNameSymDescLocationHeader}%
17014 \fi
17015 \renewcommand*\{glsgroupheading}[1]{\glslongextraGroupHeading{4}{##1}}%
17016 \renewcommand{\glossentry}[2]{%

```

```

17017 \glslongextraNameFmt{##1} &
17018 \glslongextraSymbolFmt{##1} &
17019 \glslongextraDescFmt{##1} &
17020 \glslongextraLocationFmt{##1}{##2}\tabularnewline
17021 }%
17022 \renewcommand{\subglossentry}[3]{%
17023 \glslongextraSubNameFmt{##1}{##2} &
17024 \glslongextraSubSymbolFmt{##1}{##2} &
17025 \glslongextraSubDescFmt{##1}{##2} &
17026 \glslongextraSubLocationFmt{##1}{##2}{##3}\tabularnewline
17027 }%
17028 \ifglsgnognogroupskip
17029 \renewcommand*{\glsgroupskip}{}%
17030 \else
17031 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17032 \fi
17033 }

```

mDescNameHeader

```

17034 \newcommand{\glslongextraSymDescNameHeader}{%
17035 \glslongextraSymDescNameTabularHeader\endhead
17036 \glslongextraSymDescNameTabularFooter\endfoot
17037 }

```

meTabularHeader

```

17038 \newcommand{\glslongextraSymDescNameTabularHeader}{%
17039 \toprule
17040 \glslongextraHeaderFmt\symbolname &
17041 \glslongextraHeaderFmt\descriptionname &
17042 \glslongextraHeaderFmt\entryname\tabularnewline
17043 \midrule
17044 }

```

meTabularFooter

```

17045 \newcommand{\glslongextraSymDescNameTabularFooter}{%
17046 \bottomrule
17047 }

```

g-sym-desc-name Three column style with symbol in the first column, description in the second and name in the third.

```

17048 \newglossarystyle{long-sym-desc-name}%
17049 {%
17050 \ifGlsLongExtraUseTabular
17051 \renewenvironment{theglossary}%
17052 {%
17053 \glslongextraSymSetDescWidth
17054 \edef\@glslongextra@begintab{%
17055 \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17056 \expandonce\glslongextraSymbolAlign

```

```

17057         \expandonce\glslongextraDescAlign
17058         \expandonce\glslongextraNameAlign
17059     } }%
17060     \@glslongextra@begintab
17061 }%
17062 {%
17063     \glslongextraSymDescNameTabularFooter
17064     \end{tabular}%
17065 }%
17066 \renewcommand*{\glossaryheader}{\glslongextraSymDescNameTabularHeader}%
17067 \else
17068 \renewenvironment{theglossary}%
17069 {%
17070     \glspatchLToutput
17071     \glslongextraSymSetDescWidth
17072     \edef\@glslongextra@begintab{%
17073         \noexpand\begin{longtable}{%
17074             \expandonce\glslongextraSymbolAlign
17075             \expandonce\glslongextraDescAlign
17076             \expandonce\glslongextraNameAlign
17077         } }%
17078     \@glslongextra@begintab
17079 }%
17080     {\end{longtable}}%
17081 \renewcommand*{\glossaryheader}{\glslongextraSymDescNameHeader}%
17082 \fi
17083 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{3}{##1}}%
17084 \renewcommand{\glossentry}[2]{%
17085     \glslongextraSymbolFmt{##1} &
17086     \glslongextraDescFmt{##1} &
17087     \glslongextraNameFmt{##1}\tabularnewline
17088 }%
17089 \renewcommand{\subglossentry}[3]{%
17090     \glslongextraSubSymbolFmt{##1}{##2} &
17091     \glslongextraSubDescFmt{##1}{##2} &
17092     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
17093 }%
17094 \ifglsgroupskip
17095     \renewcommand*{\glsgroupskip}{}%
17096 \else
17097     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17098 \fi
17099 }

```

mDescNameHeader

```

17100 \newcommand{\glslongextraLocationSymDescNameHeader}{%
17101     \glslongextraLocationSymDescNameTabularHeader\endhead
17102     \glslongextraLocationSymDescNameTabularFooter\endfoot
17103 }

```

meTabularHeader

```
17104 \newcommand{\glslongextraLocationSymDescNameTabularHeader}{%
17105   \toprule
17106   \glslongextraHeaderFmt\pagelistname &
17107   \glslongextraHeaderFmt\symbolname &
17108   \glslongextraHeaderFmt\descriptionname &
17109   \glslongextraHeaderFmt\entryname\tabularnewline
17110   \midrule
17111 }
```

meTabularFooter

```
17112 \newcommand{\glslongextraLocationSymDescNameTabularFooter}{%
17113   \bottomrule
17114 }
```

c-sym-desc-name Four column style with location list, symbol, description and name.

```
17115 \newglossarystyle{long-loc-sym-desc-name}%
17116 {%
17117   \ifGlsLongExtraUseTabular
17118     \renewenvironment{theglossary}%
17119     {%
17120       \glslongextraSymLocSetDescWidth
17121       \edef\@glslongextra@begintab{%
17122         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17123           \expandonce\glslongextraLocationAlign
17124           \expandonce\glslongextraSymbolAlign
17125           \expandonce\glslongextraDescAlign
17126           \expandonce\glslongextraNameAlign
17127         }}%
17128       \@glslongextra@begintab
17129     }%
17130     {%
17131       \glslongextraLocationSymDescNameTabularFooter
17132       \end{tabular}%
17133     }%
17134     \renewcommand*\glossaryheader{\glslongextraLocationSymDescNameTabularHeader}%
17135   \else
17136     \renewenvironment{theglossary}%
17137     {%
17138       \glspatchLToutput
17139       \glslongextraSymLocSetDescWidth
17140       \edef\@glslongextra@begintab{%
17141         \noexpand\begin{longtable}{%
17142           \expandonce\glslongextraLocationAlign
17143           \expandonce\glslongextraSymbolAlign
17144           \expandonce\glslongextraDescAlign
17145           \expandonce\glslongextraNameAlign
17146         }}%
17147       \@glslongextra@begintab
```

```

17148 }%
17149 {\end{longtable}}}%
17150 \renewcommand*{\glossaryheader}{\glslongextraLocationSymDescNameHeader}%
17151 \fi
17152 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{4}{##1}}%
17153 \renewcommand{\glossentry}[2]{%
17154   \glslongextraLocationFmt{##1}{##2} &
17155   \glslongextraSymbolFmt{##1} &
17156   \glslongextraDescFmt{##1} &
17157   \glslongextraNameFmt{##1}\tabularnewline
17158 }%
17159 \renewcommand{\subglossentry}[3]{%
17160   \glslongextraSubLocationFmt{##1}{##2}{##3} &
17161   \glslongextraSubSymbolFmt{##1}{##2} &
17162   \glslongextraSubDescFmt{##1}{##2} &
17163   \glslongextraSubNameFmt{##1}{##2}\tabularnewline
17164 }%
17165 \ifglsgnogroupskip
17166   \renewcommand*{\glsgroupskip}{}%
17167 \else
17168   \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17169 \fi
17170 }

```

scSymNameHeader

```

17171 \newcommand{\glslongextraDescSymNameHeader}{%
17172   \glslongextraDescSymNameTabularHeader\endhead
17173   \glslongextraDescSymNameTabularFooter\endfoot
17174 }

```

meTabularHeader

```

17175 \newcommand{\glslongextraDescSymNameTabularHeader}{%
17176   \toprule
17177   \glslongextraHeaderFmt\descriptionname &
17178   \glslongextraHeaderFmt\symbolname &
17179   \glslongextraHeaderFmt\entryname\tabularnewline
17180   \midrule
17181 }

```

meTabularFooter

```

17182 \newcommand{\glslongextraDescSymNameTabularFooter}{%
17183   \bottomrule
17184 }

```

g-desc-sym-name Three column style with description in the first column, symbol in the second and name in the third.

```

17185 \newglossarystyle{long-desc-sym-name}%
17186 {%
17187   \ifGlsLongExtraUseTabular

```

```

17188 \renewenvironment{theglossary}%
17189 {%
17190     \glslongextraSymSetDescWidth
17191     \edef\@glslongextra@begintab{%
17192         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17193             \expandonce\glslongextraDescAlign
17194             \expandonce\glslongextraSymbolAlign
17195             \expandonce\glslongextraNameAlign
17196         }}%
17197     \@glslongextra@begintab
17198 }%
17199 {%
17200     \glslongextraDescSymNameTabularFooter
17201     \end{tabular}%
17202 }%
17203 \renewcommand*\glossaryheader{\glslongextraDescSymNameTabularHeader}%
17204 \else
17205 \renewenvironment{theglossary}%
17206 {%
17207     \glspatchLToutput
17208     \glslongextraSymSetDescWidth
17209     \edef\@glslongextra@begintab{%
17210         \noexpand\begin{longtable}{%
17211             \expandonce\glslongextraDescAlign
17212             \expandonce\glslongextraSymbolAlign
17213             \expandonce\glslongextraNameAlign
17214         }}%
17215     \@glslongextra@begintab
17216 }%
17217 {\end{longtable}}%
17218 \renewcommand*\glossaryheader{\glslongextraDescSymNameHeader}%
17219 \fi
17220 \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{3}{##1}}%
17221 \renewcommand{\glossentry}[2]{%
17222     \glslongextraDescFmt{##1} &
17223     \glslongextraSymbolFmt{##1} &
17224     \glslongextraNameFmt{##1}\tabularnewline
17225 }%
17226 \renewcommand{\subglossentry}[3]{%
17227     \glslongextraSubDescFmt{##1}{##2} &
17228     \glslongextraSubSymbolFmt{##1}{##2} &
17229     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
17230 }%
17231 \ifglsgnogroupskip
17232 \renewcommand*\glsgroupskip{}%
17233 \else
17234 \renewcommand*\glsgroupskip{\glspenaltygroupskip}%
17235 \fi
17236 }

```


scSymNameHeader

```
17237 \newcommand{\glslongextraLocationDescSymNameHeader}{%
17238   \glslongextraLocationDescSymNameTabularHeader\endhead
17239   \glslongextraLocationDescSymNameTabularFooter\endfoot
17240 }
```

meTabularHeader

```
17241 \newcommand{\glslongextraLocationDescSymNameTabularHeader}{%
17242   \toprule
17243   \glslongextraHeaderFmt\pagelistname &
17244   \glslongextraHeaderFmt\descriptionname &
17245   \glslongextraHeaderFmt\symbolname &
17246   \glslongextraHeaderFmt\entryname\tabularnewline
17247   \midrule
17248 }
```

meTabularFooter

```
17249 \newcommand{\glslongextraLocationDescSymNameTabularFooter}{%
17250   \bottomrule
17251 }
```

c-desc-sym-name Four column style with location list, description, symbol and name.

```
17252 \newglossarystyle{long-loc-desc-sym-name}%
17253 {%
17254   \ifGlsLongExtraUseTabular
17255     \renewenvironment{theglossary}%
17256     {%
17257       \glslongextraSymLocSetDescWidth
17258       \edef\@glslongextra@begintab{%
17259         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17260           \expandonce\glslongextraLocationAlign
17261           \expandonce\glslongextraDescAlign
17262           \expandonce\glslongextraSymbolAlign
17263           \expandonce\glslongextraNameAlign
17264         }}%
17265       \@glslongextra@begintab
17266     }%
17267     {%
17268       \glslongextraLocationDescSymNameTabularFooter
17269       \end{tabular}%
17270     }%
17271     \renewcommand*\glossaryheader{\glslongextraLocationDescSymNameTabularHeader}%
17272   \else
17273     \renewenvironment{theglossary}%
17274     {%
17275       \glspatchLToutput
17276       \glslongextraSymLocSetDescWidth
17277       \edef\@glslongextra@begintab{%
17278         \noexpand\begin{longtable}{%

```

```

17279         \expandonce\glslongextraLocationAlign
17280         \expandonce\glslongextraDescAlign
17281         \expandonce\glslongextraSymbolAlign
17282         \expandonce\glslongextraNameAlign
17283     } }%
17284     \@glslongextra@begintab
17285 }%
17286 {\end{longtable}}%
17287 \renewcommand*{\glossaryheader}{\glslongextraLocationDescSymNameHeader}%
17288 \fi
17289 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{4}{##1}}%
17290 \renewcommand{\glossentry}[2]{%
17291     \glslongextraLocationFmt{##1}{##2} &
17292     \glslongextraDescFmt{##1} &
17293     \glslongextraSymbolFmt{##1} &
17294     \glslongextraNameFmt{##1}\tabularnewline
17295 }%
17296 \renewcommand{\subglossentry}[3]{%
17297     \glslongextraSubLocationFmt{##1}{##2}{##3} &
17298     \glslongextraSubDescFmt{##1}{##2} &
17299     \glslongextraSubSymbolFmt{##1}{##2} &
17300     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
17301 }%
17302 \ifglsnogroupskip
17303     \renewcommand*{\glsgroupskip}{}%
17304 \else
17305     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17306 \fi
17307 }

```

5 topic styles (glossary-topic.sty)

5.1 Package Initialisation and Options

Provides “topic” styles where top-level entries are considered a topic.

```
17308 \NeedsTeXFormat{LaTeX2e}
```

```
17309 \ProvidesPackage{glossary-topic}[2021/09/20 v1.46 (NLCT)]
```

Load required package.

```
17310 \RequirePackage{multicol}
```

The top-level entries act like headers. If the top-level entry has a description it’s placed below the name.

topic

```
17311 \newglossarystyle{topic}{%
```

```
17312   \renewenvironment{theglossary}{%
```

```
17313   {%
```

```
17314     \glstopicInit
```

```
17315     \def\glstopic@prechildren{}}%
```

```
17316     \def\glstopic@prevlevel{-1}%
```

```
17317   }%
```

```
17318   {\par}%
```

```
17319   \renewcommand*{\glossaryheader}{}%
```

```
17320   \renewcommand*{\glsgroupheading}[1]{%
```

```
17321     \def\glstopic@prevlevel{-1}%
```

```
17322     \glstopicGroupHeading{##1}%
```

```
17323   }%
```

```
17324   \renewcommand{\glossentry}[2]{%
```

```
17325     \hangindent0pt\relax
```

```
17326     \parindent\glstopicParIndent\relax
```

```
17327     \glstopicItem{##1}{##2}%
```

If there isn’t a description, penalise a page break.

```
17328     \ifglshasdesc{##1}%
```

```
17329     {%
```

```
17330       \def\glstopic@prechildren{}}%
```

```
17331     }%
```

```
17332     {%
```

```
17333       \def\glstopic@prechildren{\nopagebreak}%
```

```
17334     }%
```

```
17335   }%
```

```
17336   \renewcommand{\subglossentry}[3]{%
```

```
17337     \ifnum\glstopic@prevlevel=0\relax\glstopic@prechildren\fi
```

```
17338     \def\glstopic@prevlevel{##1}%
```

Grouping is added to scope the effect of `\everypar`.

```

17339 \begingroup
17340 \glstopicAssignSubIndent{##1}%
17341 \glstopicSubItem{##1}{##2}{##3}%

17342 \par
17343 \endgroup
17344 }%
17345 \renewcommand*{\glsgroupskip}{}%
17346 }

```

`\picGroupHeading`

`\glstopicGroupHeading{<group label>}`

May be redefined if letter group headings are required. For example:

```

\renewcommand*{\glstopicGroupHeading}[1]{%
  \glstrgetgrouptitle{#1}{\thisgrptitle}%
  \section*{\thisgrptitle}%
}

17347 \newcommand*{\glstopicGroupHeading}[1]{%

```

`\glstopicItem`

`\glstopicItem{<label>}{<location list>}`

```

17348 \newcommand*{\glstopicItem}[2]{%
17349 \glspare\glstopicPreSkip\glspare\noindent
17350 \glstopicMarker{#1}%
17351 \glstopicTitleFont
17352 {%
17353 \glstryitem{#1}\glstarget{#1}{\glstopicTitle{#1}}%
17354 }%
17355 \ifglshasdesc{#1}%
17356 {\glspare\nobreak\glstopicMidSkip\glspare\nobreak
17357 \@afterheading\glstopicDesc{#1}\glspare\glstopicPostSkip}%
17358 {\glspare\nobreak\glstopicPostSkip}%
17359 \glstopicLoc{#1}{#2}%
17360 }

```

`\glstopicMarker` May be used to insert a bookmark etc if required.

```

17361 \newcommand*{\glstopicMarker}[1]{%

```

`\glstopicName`

```

17362 \newcommand*{\glstopicTitle}[1]{\Glossentryname{#1}%
17363 \ifglshassymbol{#1}{\space\glossentrysymbol{#1}}{}}%
17364 }

```

stopicTitleFont

```
17365 \newcommand*{\glstopicTitleFont}[1]{\textbf{\large #1}}
```

\glstopicDesc

```
17366 \newcommand*{\glstopicDesc}[1]{\Glossentrydesc{#1}\glspostdescription}
```

\glstopicLoc

```
17367 \newcommand*{\glstopicLoc}[2]{}
```

stopicParIndent

```
17368 \newlength\glstopicParIndent
```

```
17369 \setlength\glstopicParIndent{20pt}
```

stopicSubIndent

```
17370 \newlength\glstopicSubIndent
```

```
17371 \setlength\glstopicSubIndent{20pt}
```

\glstopicInit

```
17372 \newcommand{\glstopicInit}{}%
```

AssignSubIndent

```
\glstopicAssignSubIndent{<level>}
```

Used to set the indentation for sub-levels.

```
17373 \newcommand*{\glstopicAssignSubIndent}[1]{%
```

```
17374   \par
```

```
17375   \parindent\dimexpr#1\glstopicSubIndent-\glstopicSubIndent\relax
```

```
17376   \glstopicAssignWidest{#1}%
```

```
17377   \glstopicsubitemhangindent=\dimexpr\parindent+\glstopicwidest\relax
```

```
17378   \hangindent\glstopicsubitemhangindent\relax
```

```
17379   \everypar{\hangindent\glstopicsubitemhangindent\relax
```

```
17380     \parindent\dimexpr\glstopicSubItemParIndent+\glstopicsubitemhangindent\relax}%
```

```
17381 }%
```

bitemhangindent

```
17382 \newlength\glstopicsubitemhangindent
```

ubItemParIndent

```
17383 \newlength\glstopicSubItemParIndent
```

```
17384 \glstopicSubItemParIndent\parindent
```

\glstopicwidest

```
17385 \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.

```
17386 \newcommand*{\glstopicAssignWidest}[1]{%
17387   \ifcsundef{@glswidestlength\romannumeral#1}%
17388   {%
17389     \ifcsdef{@glswidestname\romannumeral#1}%
17390     {%
17391       \settowidth{\glstopicwidest}{%
17392         \glstopicSubNameFont{\csuse{@glswidestname\romannumeral#1}}%
17393         \glstopicSubItemSep
17394       }%
17395     }%
17396     {\setlength{\glstopicwidest}{0pt}}%
```

Save the value so that it doesn't have to keep being recalculated.

```
17397   \csedef{@glswidestlength\romannumeral#1}{\the\glstopicwidest}%
17398   }%
17399   {\setlength{\glstopicwidest}{\csuse{@glswidestlength\romannumeral#1}}}%
17400 }
```

glstopicPreSkip

```
17401 \newcommand*{\glstopicPreSkip}{\medskip}
```

glstopicMidSkip

```
17402 \newcommand*{\glstopicMidSkip}{\smallskip}
```

lstopicPostSkip

```
17403 \newcommand*{\glstopicPostSkip}{\smallskip}
```

glstopicSubItem

`\glstopicSubItem{<level>}{<label>}{<location list>}`

```
17404 \newcommand*{\glstopicSubItem}[3]{%
17405   \glstopicSubItemBox{#1}{\glstopicSubNameFont{\glentryitem{#2}}%
17406     \glstarget{#2}{\glossentryname{#2}}}%
17407   \glstopicSubItemSep
17408   }%
17409   \ifglshassymbol{#2}{(\glossentrysymbol{#2})\space}{}%
17410   \ifglshasdesc{#2}%
17411   {\glossentrydesc{#2}\glspostdescription\glstopicSubPreLocSep}{}%
17412   \glstopicSubLoc{#2}{#3}%
17413 }
```

topicSubItemSep

```
17414 \newcommand*{\glstopicSubItemSep}{\quad}
```

topicSubItemBox

```
\glstopicSubItemBox{<level>}{<text>}
```

```
17415 \newcommand*{\glstopicSubItemBox}[2]{%
```

```
17416   \ifdim\glstopicwidest>0pt\relax\makebox[\glstopicwidest][l]{#2}\else#2\fi
```

```
17417 }
```

opicSubNameFont

```
17418 \newcommand*{\glstopicSubNameFont}[1]{\textbf{#1}}
```

picSubPreLocSep

```
17419 \newcommand*{\glstopicSubPreLocSep}{\space}
```

\glstopicSubLoc

```
17420 \newcommand*{\glstopicSubLoc}[2]{#2}
```

\glstopicCols

```
17421 \newcommand*{\glstopicCols}{2}
```

glstopicColsEnv

```
17422 \newcommand*{\glstopicColsEnv}{multicols}
```

topicmcols

```
17423 \newglossarystyle{topicmcols}{%
```

```
17424   \renewenvironment{theglossary}{%
```

```
17425   {%
```

```
17426     \glstopicInit
```

```
17427     \def\glstopic@prechildren{}%
```

```
17428     \def\glstopic@postchildren{}%
```

```
17429     \def\glstopic@prevlevel{-1}%
```

```
17430   }%
```

```
17431   {%
```

```
17432     \ifnum\glstopic@prevlevel>0\relax\glstopic@postchildren\fi
```

```
17433     \par
```

```
17434   }%
```

```
17435   \renewcommand*{\glossaryheader}{}%
```

```
17436   \renewcommand*{\glsgroupheading}[1]{%
```

```
17437     \ifnum\glstopic@prevlevel>0\relax\glstopic@postchildren\fi
```

```
17438     \def\glstopic@prevlevel{-1}%
```

```
17439     \glstopicGroupHeading{##1}%
```

```
17440   }%
```

```
17441   \renewcommand{\glossentry}[2]{%
```

```

17442 \ifnum\glstopic@prevlevel>0\relax\glstopic@postchildren\fi
17443 \def\glstopic@prevlevel{0}%
17444 \hangindentOpt\relax
17445 \parindent\glstopicParIndent\relax
17446 \glstopicItem{##1}{##2}%
17447 \ifnum\glstopicCols>1\relax

```

If there isn't a description, penalise a page break.

```

17448 \ifglshasdesc{##1}%
17449 {%
17450 \edef\glstopic@prechildren{%
17451 \noexpand\begin{\glstopicColsEnv}{\glstopicCols}%
17452 }%
17453 }%
17454 {%
17455 \edef\glstopic@prechildren{%
17456 \noexpand\nopagebreak
17457 \noexpand\begin{\glstopicColsEnv}{\glstopicCols}%
17458 }%
17459 }%
17460 \edef\glstopic@postchildren{\noexpand\end{\glstopicColsEnv}}%
17461 \fi
17462 }%
17463 \renewcommand{\subglossentry}[3]{%
17464 \ifnum\glstopic@prevlevel=0\relax\glstopic@prechildren\fi
17465 \def\glstopic@prevlevel{##1}%
17466 \glstopicAssignSubIndent{##1}%
17467 \glstopicSubItem{##1}{##2}{##3}%
17468 }%
17469 \renewcommand*{\glsgroupskip}{}%
17470 }

```


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	87
General: Initial experimental release	5	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	87
\Glsfmtshort: new	367	\@Glstext@: added redefinition	83
\glsfmtshort: new	366	\@Glsuseri@: added redefinition	88
\Glsfmtshortpl: new	367	\@Glsuserii@: added redefinition	88
\glsfmtshortpl: new	366	\@Glsuseriii@: added redefinition	88
short: switched inline full form to short		\@Glsuseriv@: added redefinition	88
(long)	258	\@Glsuserv@: added redefinition	89
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	89
\@ACRlong: added redefinition	93	\@acrlong: added redefinition	92
\@ACRlongpl: added redefinition	94	\@acrlongpl: added redefinition	93
\@ACRshort: added redefinition	90	\@acrshort: added redefinition	90
\@ACRshortpl: added redefinition	92	\@acrshortpl: added redefinition	91
\@Acrlong: added redefinition	92	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	93	argument	74
\@Acrshort: added redefinition	90	\@glsdescplural@: added redefinition	86
\@Acrshortpl: added redefinition	91	\@glsfirst@: added redefinition	83
\@GLSdesc@: added redefinition	86	\@glsfirstplural@: added redefinition	84
\@GLSdescplural@: added redefinition	86	\@glsplural@: added redefinition	84
\@GLSfirst@: added redefinition	83	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	85	redefinition	87
\@GLSname@: added redefinition	85	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	84	new	151
\@GLSSymbol@: added redefinition	87	\@glsxtr@field@linkdefs: new	82
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	223
redefinition	87	\@print@glossary: added redefinition	148
\@GLStext@: added redefinition	83	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	88	\abbrvdefaultfont	229
\@GLSuserii@: added redefinition	88	\glsaccessdesc: new	182
\@GLSuseriii@: added redefinition	88	\glsaccessdescplural: new	182
\@GLSuseriv@: added redefinition	89	\glsaccessfirst: new	179
\@GLSuserv@: added redefinition	89	\glsaccessfirstplural: new	180
\@GLSuservi@: added redefinition	89	\Glsaccesslong: new	184
\@GLSdesc@: added redefinition	86	\glsaccesslong: new	184
\@GLSdescplural@: added redefinition	86	\glsaccessname: new	178
\@Glsfirst@: added redefinition	83	\glsaccessplural: new	179
\@Glsfirstplural@: added redefinition	85	\Glsaccessshort: new	183
\@Glsname@: added redefinition	85	\glsaccessshort: new	183
\@Glsplural@: added redefinition	84	\Glsaccessshortpl: new	184

\glsaccesssshortpl: new	184	\cGLSpl: new	121
\glsaccessssymbol: new	181	\cGLSpl@: new	121
\glsaccessssymbolplural: new	181	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	178	new	116
\glsentryfmt: added check for short ..	73	\cGLS: new	121
\glslongpltok: new	223	\cGLSformat: new	121
\glsshortpltok: new	223	\cGLSpl: new	121
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	121
name in \setkeys	224	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	19
for plural	220	\glsenableentrycount: new	116
\GLSxtrlongpl: new	239	\glsfirstabbrvdefaultfont: new ..	229
\Glsxtrlongpl: new	239	\glsfirstlongdefaultfont: new ...	229
\glsxtrlongpl: new	238	\Glsfmtfirst: new	370
\glsxtrNoGlossaryWarning: new	25	\glsfmtfirst: new	370
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	371
new	219	\glsfmtfirstpl: new	371
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	369
new	219	\glsfmtplural: new	369
\glsxtrpostlinkendsentence: new ..	219	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	238	\Glsxtrtitleshort	367
\Glsxtrshortpl: new	237	renamed from \Glsentryfmtshort ..	367
\glsxtrshortpl: new	236	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort	366
\glslabeltok	250	renamed from \glsentryfmtshort ..	366
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	248	\Glsxtrtitleshortpl	367
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glsentryfmtshortpl	367
redefinition of \acronymtype	21	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	366
\Glsxtrshort	367	renamed from	
\glsfmtshort: changed to use		\glsentryfmtshortpl	366
\glsxtrshort	366	\Glsfmttext: new	368
\Glsfmtshortpl: changed to use		\glsfmttext: new	368
\glsxtrshortpl	367	\glshasattribute: new	196
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	196
\glsxtrshortpl	366	\glsxtremsuffix: new	299
\glsxtrifemptyglossary: new	33	\GlsXtrEnableEntryCounting: new ..	116
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new	118
argument	200	\glsxtrscfont: new	266
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new	266
argument	200	\glsxtrsmfont: new	282
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new	283
default type to \acronymtype	131	short-em: new	307
\newterm: fixed name argument	199	short-em-desc: new	309
0.5 (2015-12-07)		short-em-footnote: new	318
\cGLS: new	121	short-em-long: new	303
\cGLS@: new	121	short-em-long-desc: new	305

short-em-postfootnote: new	321	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new	277	attribute	356
short-sc-postfootnote: new	280	\Glsxtrheadtext: now uses headuc	
short-sm: new	287	attribute	359
short-sm-desc: new	288	\glxtrheadtext: now uses headuc	
short-sm-footnote: new	294	attribute	359
short-sm-long: new	285	short-em-footnote: switch off regular	
short-sm-long-desc: new	286	attribute if set	319
short-sm-postfootnote: new	296	short-em-footnote-desc: switch off	
long-noshort-em: new	311	regular attribute if set	321
long-noshort-em-desc: new	315	short-long: switch off regular attribute	
long-noshort-sm: new	290	if set	249
long-noshort-sm-desc: new	292	short-long-desc: switch off regular	
long-short-em: new	299	attribute if set	251
long-short-em-desc: new	301	short-postfootnote-desc: switch off	
long-short-sm: new	283	regular attribute if set	257
long-short-sm-desc: new	284	short-sc-footnote: switch off regular	
0.5.1 (2015-12-02)		attribute if set	277
\Glsaccessstext: new	179	short-sc-footnote-desc: switch off	
0.5.1 (2015-12-07)		regular attribute if set	279
\@glxtr@doaccsupp: new	24	short-sm-footnote: switch off regular	
General: removed \ifglxtruseuchead	356	attribute if set	294
\Glsaccessdesc: new	182	short-sm-footnote-desc: switch off	
\Glsaccessdescplural: new	183	regular attribute if set	296
\Glsaccessfirst: new	180	long-short: switch off regular attribute	
\Glsaccessfirstplural: new	180	if set	247
\Glsaccessname: new	178	long-short-desc: switch off regular	
\Glsaccessplural: new	179	attribute if set	248
\Glsaccesssymbol: new	181	long-short-sc-desc: switch off regular	
\Glsaccesssymbolplural: new	181	attribute if set	268
\Glsxtrheadfirst: now uses headuc		footnote: switch off regular attribute if	
attribute	361	set	252
\glxtrheadfirst: now uses headuc		postfootnote: switch off regular	
attribute	360	attribute if set	255
\Glsxtrheadfirstplural: now uses		0.5.2 (2015-12-08)	
headuc attribute	362	\@GLSdesc@: added accessibility support	86
\glxtrheadfirstplural: now uses		\@GLSdescplural@: added accessibility	
headuc attribute	361	support	86
\Glsxtrheadplural: now uses headuc		\@GLSfirst@: added accessibility	
attribute	360	support	83
\glxtrheadplural: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	359	support	85
\Glsxtrheadshort: now uses headuc		\@GLSname@: added accessibility support	85
attribute	357	\@GLSplural@: added accessibility	
\glxtrheadshort: now uses headuc		support	84
attribute	356	\@GLSsymbol@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support	87
attribute	357	\@GLSsymbolplural@: added	
		accessibility support	87

\@GLStext@: added accessibility support	83	\GLSaccesslongpl: new	185, 194
\@Glsdesc@: added accessibility support	86	\Glsaccesslongpl: new	185
\@Glsdescplural@: added accessibility support	86	\glsaccesslongpl: new	185
\@Glsfirst@: added accessibility support	83	\GLSaccessname: new	178, 192
\@Glsfirstplural@: added accessibility support	85	\GLSaccessplural: new	179, 192
\@Glsname@: add accessibility support	85	\GLSaccessshort: new	183, 194
\@Glsplural@: added accessibility support	84	\GLSaccessshortpl: new	184, 194
\@Glsymbol@: added accessibility support	87	\GLSaccesssymbol: new	181, 193
\@Glsymbolplural@: added accessibility support	87	\GLSaccesssymbolplural: new	182, 193
\@Glstext@: added accessibility support	83	\GLSaccessstext: new	179, 192
\@Glsdesc@: added accessibility support	86	\glsentryfmt: moved	
\@Glsdescplural@: added accessibility support	86	\glssetabbrvfmt from	
\@Glsfirst@: added accessibility support	83	\glsxtrabbrvfmt to here	73
\@Glsfirstplural@: added accessibility support	84	\GlsXtrEnableInitialTagging: new	215
\@Glsname@: added accessibility support	85	\glsxtrfieldtitlecase: new	201
\@Glsplural@: added accessibility support	84	\GlsXtrFormatLocationList: new	71
\@Glsymbol@: added accessibility support	87	\glsxtrnewabbrevpresetkeyhook:	
\@Glsymbolplural@: added accessibility support	87	new	227
\@Glstext@: added accessibility support	82	\glsxtrtagfont: new	217
\@glsxtr@activate@initialtagging:		\KV@printgloss@nonumberlist: added	73
new	216	\mfu@checkword@do: added	216
\@glsxtr@do@titlecaps@warn: new	216	\setabbreviationstyle: added check	
\@glsxtr@tag: new	217	for post-definition style switch	243
General: fixed typo in glossaries-accsupp		0.5.3 (2015-12-09)	
and tidied up code to use just one		\@glsxtr@autoindex@at: new	212
\@ifpackageloaded	178	\@glsxtr@autoindex@encap: new	212
removed \glsxtrabbrvfmt	240	\@glsxtr@autoindex@esc: new	212
\glossaryentrynumbers: added	71	\@glsxtr@autoindex@level: new	212
\Glossentrydesc: added	214	\@glsxtr@autoindex@setname: new	210
\Glossentryname: added	205	\@glsxtr@doabbreviationsdef: new	21
\Glossentrysymbol: added	215	General: removed	
\glossentrysymbol: added	214	\GlsXtrNoGlsWarningNoAutoMakeMain	150
\GLSaccessdesc: new	182, 193	\glsdescwidth: added	70
\GLSaccessdescplural: new	183, 193	\glspagelistwidth: added	70
\GLSaccessfirst: new	180, 192	\glsxtrdoautoindexname: new	209
\GLSaccessfirstplural: new	180, 192	\glsxtrpostnamehook: new	207
\GLSaccesslong: new	184, 194	\if@glsxtr@format@override: new	209
		\ProvidesGlossariesExtraLang: new	375
		\RequireGlossariesExtraLang: new	375
		0.5.4 (2015-12-15)	
		\@newglossaryentry@defunitcounters:	
		new	122
		\@GLSxtr@p@acrlong@: new	107
		\@GLSxtr@p@acrlongpl@: new	107
		\@GLSxtr@p@acrshort@: new	107
		\@GLSxtr@p@acrshortpl@: new	107
		\@GLSxtr@p@long@: new	106
		\@GLSxtr@p@longpl@: new	107

\@GLSxtr@p@plural@: new	105	\@sGlsXtrEnableOnTheFly: new	66
\@GLSxtr@p@short@: new	106	\cGlsformat: added	122
\@GLSxtr@p@shortpl@: new	106	\cGlsformat: added	121
\@GLSxtr@p@text@: new	105	\cGlsplformat: added	122
\@GlsXtrEnableOnTheFly: new	66	\cGlsplformat: added	122
\@Glsxtr: new	67	\gl:disablehyper: added	103
\@Glsxtr@p@acrlong@: new	107	\gl:donohyperlink: added	104
\@Glsxtr@p@acrlongpl@: new	107	\gl:enableentryunitcount: new ...	124
\@Glsxtr@p@acrshort@: new	107	\gl:shasattribute: added check for	
\@Glsxtr@p@acrshortpl@: new	107	entry's existence	196
\@Glsxtr@p@long@: new	106	\gl:sifattribute: added check for	
\@Glsxtr@p@longpl@: new	106	entry's existence	197
\@Glsxtr@p@plural@: new	105	\gl:postlinkhook: added existence	
\@Glsxtr@p@short@: new	105	check	218
\@Glsxtr@p@shortpl@: new	106	\Glsxtr: new	67
\@Glsxtr@p@text@: new	105	\gl:xtr: new	67
\@Glsxtrpl: new	68	\gl:xtrcat: new	67
\@alt@gl:shyp@opt: new	101	\gl:xtrdohyperlink: added	102
\@gl:alt@hyp@opt: new	100	\gl:xtrdowrglossaryhook: new	100
\@gl:alt@hyp@opt@char: new	101	\GlsXtrEnableEntryUnitCounting:	
\@gl:alt@hyp@opt@keys: new	101	new	127
\@gl:increment@currunitcount:		\GlsXtrEnableOnTheFly: new	66
new	123	\Glsxtrpl: new	68
\@gl:local@increment@currunitcount:		\gl:xtrpl: new	67
new	123	\gl:xtrpostlocalreset: new	115
\@gl:setdefault@gl:link@opts:		\gl:xtrpostlocalunset: new	115
new	98	\gl:xtrpostreset: new	115
\@gl:xtr: new	67	\gl:xtrpostunset: new	113
\@gl:xtr@addunitcounter: new	122	\gl:xtrprotectlinks: new	104
\@gl:xtr@currunitcount: new	124	\GlsXtrSetAltModifier: new	101
\@gl:xtr@ifunitcounter: new	123	\GlsXtrSetDefaultGlsOpts: new	99
\@gl:xtr@p@acrlong@: new	107	\gl:xtrstarflywarn: new	66
\@gl:xtr@p@acrlongpl@: new	107	\GlsXtrWarning: new	68
\@gl:xtr@p@acrshort@: new	107	\MakeAcronymsAbbreviations: now	
\@gl:xtr@p@acrshortpl@: new	107	disables \setacronymstyle	131
\@gl:xtr@p@long@: new	106	1.0 (2016-01-24)	
\@gl:xtr@p@longpl@: new	106	\@gl:xtr@autoindexcrossrefs: new .	18
\@gl:xtr@p@plural@: new	105	\@gl:xtr@idx@displaynumberlist:	
\@gl:xtr@p@short@: new	105	new	142
\@gl:xtr@p@shortpl@: new	106	\@gl:xtr@idx@entrynumberlist: new	143
\@gl:xtr@p@text@: new	105	\@gl:xtr@noidx@displaynumberlist:	
\@gl:xtr@prevunitcount: new	124	new	142
\@gl:xtr@setentryunitcountunsetattr:		\@gl:xtr@noidx@entrynumberlist:	
new	128	new	143
\@gl:xtr@unitcountlist: new	122	\@gl:xtr@noidx@numberlistloop:	
\@gl:xtrpl: new	67	new	142
\@newglossaryentryposthook: added		\@gl:xtr@reg@glosslist: new	132
empty see value if not set and added		\makeglossaries: new	133
'see' to field key map	55		

1.01 (2016-02-02)	
\glxtrdiscardperiod: added check for first use	220
short-desc: fixed typo in \glxtrininlinefullformat and added missing second argument ...	260
1.02 (2016-04-25)	
\@glxtr@current@style: new	69
\Glsfmtfull: new	374
\Glsfmtfull: new	373
\Glsfmtfullpl: new	375
\Glsfmtfullpl: new	374
\Glsfmtlong: new	372
\Glsfmtlong: new	372
\Glsfmtlongpl: new	373
\Glsfmtlongpl: new	372
\Glsxtrheadfull: new	365
\Glsxtrheadfullpl: new	364
\Glsxtrheadfullpl: new	365
\Glsxtrheadlong: new	363
\Glsxtrheadlong: new	362
\Glsxtrheadlongpl: new	364
\Glsxtrheadlongpl: new	363
\Glsxtrtitlefull: new	365
\Glsxtrtitlefull: new	364
\Glsxtrtitlefullpl: new	366
\Glsxtrtitlefullpl: new	365
\Glsxtrtitlelong: new	363
\Glsxtrtitlelong: new	362
\Glsxtrtitlelongpl: new	364
\Glsxtrtitlelongpl: new	363
short-postfootnote-desc: added redef of \glxtrsetupfulldefs ..	257
\ifglxtrininsertinside: new	246
postfootnote: added redef of \glxtrsetupfulldefs	255
stylemods: new	25
1.03 (2016-04-27)	
\@GLSfirstplural@: bug fix: misspelt cs name	85
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	84
\@GLSfirstplural@: bug fix: misspelt cs name	85
\@GLsplural@: fixed bug \@GLsplural@ should be redefined not \@GLsplural	84
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	84
\glxtrtitlelongpl: bug fix: changed \glxtrlong to \glxtrlongpl ..	363
\glxtrtitleshortpl: bug fix: changed \glxtrshort to \glxtrshortpl	356
1.04 (2015-04-30)	
short-em-footnote: renamed from “footnote-em”	318
1.04 (2016-05-02)	
\@@glxtrpostloctag: new	72
\@GLSdesc@: set abbreviation and regular format	86
\@GLSdescplural@: set abbreviation and regular format	86
\@GLSfirst@: set abbreviation format ..	83
\@GLSfirstplural@: set abbreviation and regular format	85
\@GLSname@: set abbreviation and regular format	85
\@GLSplural@: set abbreviation and regular format	84
\@GLSsymbol@: set regular format	87
\@GLSsymbolplural@: set regular format	87
\@GLStext@: set abbreviation and regular format	83
\@GLSuseri@: set regular format	88
\@GLSuserii@: set regular format	88
\@GLSuseriii@: set regular format	88
\@GLSuseriv@: set regular format	89
\@GLSuseriv@: set regular format	89
\@GLSuseriv@: set regular format	89
\@GLSdesc@: set abbreviation and regular format	86
\@GLSdescplural@: set abbreviation and regular format	86
\@GLSfirst@: set abbreviation and regular format	83
\@GLSfirstplural@: set abbreviation and regular format	85
\@GLSname@: set abbreviation and regular format	85
\@GLsplural@: set abbreviation and regular format	84
\@GLssymbol@: set regular format	87
\@GLssymbolplural@: set regular format	87
\@GLStext@: set abbreviation and regular format	83
\@GLSuseri@: set regular format	88
\@GLSuserii@: set regular format	88
\@GLSuseriii@: set regular format	88

\@Glsuseriv@: set regular format	88	\glslongfont: new	229
\@Glsuserv@: set regular format	89	\glslonguserfont: new	325
\@Glsuservi@: set regular format	89	\glstrassignfieldfont: new	82
\@gls@preglossaryhook: added check for entry's existence	217	\GlsXtrEnablePreLocationTag: new	71
\@glsdesc@: set abbreviation and regular format	86	\glstrfirstscfont: new	266
\@glsdescplural@: set abbreviation and regular format	86	\glstrfirstsmfont: new	283
\@glsfirst@: set abbreviation and regular format	83	\glstrlongshortdescsort: new	248
\@glsfirstplural@: set abbreviation and regular format	84	\glstrpostnamehook: added category check	207
\@glsname@: set abbreviation and regular format	85	\glstrregularfont: new	73
\@glsplural@: set abbreviation and regular format	84	\glstruserfield: new	324
\@glsymbol@: set regular format	87	\glstruserparen: new	324
\@glsymbolplural@: set regular format	87	\glstrusersuffix: new	325
\@glstext@: set abbreviation and regular format	82	\GlsXtrWarnDeprecatedAbbrStyle: new	245
\@glstr@deprecated@abbrstyle: new	245	short-em-long-em: new	305
\@glstr@do@style: new	26	short-em-long-em-desc: new	307
\@glstr@dolocktag: new	73	short-em-nolong: new	309
\@glstr@idx@entrynumberlist: switched from \let to \newcommand	143	short-em-nolong-desc: new	310
\@glstr@pagetag: new	72	short-em-postfootnote: renamed from "postfootnote-em"	321
\@glstr@pagetag: new	72	short-footnote: new	253
\@glstr@prelocktag: new	73	short-long-user: new	332
\@glstr@postlocktag: new	72	short-long-user-desc: new	333
\@glstr@prelocktag: new	72	short-nolong: new	259
\glossentrydesc: added glossdescfont attribute check	201	short-nolong-desc: new	261
\Glossentryname: added glossnamefont attribute check	205	short-postfootnote: new	256
\glossentryname: added glossnamefont attribute check	203	short-sc-footnote: renamed from "footnote-sc"	277
moved post name hook inside condition	205	short-sc-nolong: new	272
\glsabbrvmfont: new	299	short-sc-nolong-desc: new	273
\glsabbrvuserfont: new	324	short-sc-postfootnote: renamed from "postfootnote-sc"	280
\glsfirstabbrvmfont: new	299	short-sm-footnote: renamed from "footnote-sm"	294
\glsfirstabbrvuserfont: new	324	short-sm-nolong: new	288
\glsfirstlongemfont: new	299	short-sm-nolong-desc: new	290
\glsfirstlonguserfont: new	325	short-sm-postfootnote: renamed from "postfootnote-sm"	296
\glsifnotregularcategory: new	198	\letabbreviationstyle: new	245
\glslongdefaultfont: new	229	\newabbreviationstyle: bug fix: corrected test for existence	244
\glslongemfont: new	299	long-em-noshort-em: new	313
		long-em-noshort-em-desc: new	316
		long-em-short-em: new	301
		long-em-short-em-desc: new	303
		long-noshort: new	265
		long-noshort-desc: new	264

long-noshort-em: renamed from “long-em”	311	\glxtrComputeTreeIndent: new ...	451
long-noshort-em-desc: renamed from “long-desc-em”	315	\glxtrComputeTreeSubIndent: new	451
long-noshort-sc: renamed from “long-sc”	274	\glxtrtreetopindent: new	442
long-noshort-sc-desc: renamed from “long-desc-sc”	275	short-em-long: fixed incorrect font used by long form	304
long-noshort-sm: renamed from “long-sm”	290	\xglsssetwidest: new	443
long-noshort-sm-desc: renamed from \long-desc-sm	292	1.06 (2016-06-18)	
long-short-user: new	325	\@glstdoifexistsorwarn: new	18
long-short-user-desc: new	331	\@glxtr@docdefval: new	17
\renewabbreviationstyle: new	245	\@glxtr@usesee: new	55
style: new	26	General: disabled docdef key at the start of the document	32
1.05 (2016-06-10)		docdef option changed to choice	17
\eglssetwidest: new	443	\glxtr@usesee: new	55
\glsFindWidestAnyName: new	445	\glxtrusesee: new	55
\glsFindWidestAnyNameLocation: new	451	\glxtruseseeformat: new	55
\glsFindWidestAnyNameSymbol: new	448	\if@glxtrdocdefrestricted: new ..	18
\glsFindWidestAnyNameSymbolLocation: new	449	1.07 (2016-08-15)	
\glsFindWidestLevelTwo: new	447	\@@glxtrp: new	108
\glsFindWidestUsedAnyName: new ..	445	\@GLSfirst@: added check for nohyperfirst attribute	84
\glsFindWidestUsedAnyNameLocation: new	450	\@GLSfirstplural@: added check for nohyperfirst attribute	85
\glsFindWidestUsedAnyNameSymbol: new	448	\@GLSxtrp: new	109
\glsFindWidestUsedAnyNameSymbolLocation: new	449	\@GLSfirst@: added check for nohyperfirst attribute	83
\glsFindWidestUsedLevelTwo: new .	446	\@GLSfirstplural@: added check for nohyperfirst attribute	85
\glsFindWidestUsedTopLevelName: new	444	\@Glsxtrp: new	108
\glsfirstlongfootnotefont: new ..	251	\@gls@preglossaryhook: added \glossxtrsetpopts	217
\glsgetwidestname: new	444	\@glsfirst@: added check for nohyperfirst attribute	83
\glsgetwidestsubname: new	444	\@glsfirstplural@: added check for nohyperfirst attribute	84
\glslongfootnotefont: new	251	\@glsxtrinmark: new	353
\glxtrAltTreeIndent: new	442	\@glsxtrnotinmark: new	354
\glxtralttreeInit: new	442	\@glsxtrp: new	108
\glxtrAltTreePar: new	442	\@glsxtrp@opt: new	107
\glxtrAltTreeSetHangIndent: new	452	\glossxtrsetpopts: new	108
\glxtrAltTreeSetSubHangIndent: new	452	\glsps: new	110
\glxtralttreeSubSymbolDescLocation: new	442	\glspt: new	110
\glxtralttreeSymbolDescLocation: new	441	\glxtr@entry@p: new	109
		\glxtrabbrvfootnote: new	251
		\glxtrchecknohyperfirst: new	83
		\glxtrfieldtitlecasecs: new	201
		\glxtrifinmark: new	353
		\GLSxtrp: new	111
		\Glsxtrp: new	110

\glxtrp: new	109	\glxtrassignfieldfont: added check for existence	82
\glxtrsetpopts: new	108	\glxtrresourcefile: new	153
short-long-desc: added text key	251	\printunsrtglossaries: new	160
fixed misspelling of \glsabbrvfont in plural key	251	\printunsrtglossary: new	159
long-short-desc: added missing text key	248	1.09 (2016-12-16)	
fixed misspelling of \glsabbrvfont .	248	\@glxtr@gettype: new	141
footnote: changed first forms to use \glsfirstlongfootnotefont ...	252	\@glxtr@mixed@assign@sortkey: new	141
postfootnote: removed \footnote from first keys	254	\@printglossary: redefined to save options	139
switched from \glsfirstlongfont to \glsfirstlongfootnotefont ...	256	\glxtr@makeglossaries: new	141
\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glxtrifwasfirstuse	132	1.10 (2016-12-17)	
1.08 (2016-12-13)		\@GLSpl@: fixed bug caused by typo in command name	75
\@@glxtr@record: new	8	1.11 (2017-01-19)	
\@GLS@: added \@glxtr@record	75	\@glxtr@do@redef@forglsentries: new	6
\@GLSpl@: added \@glxtr@record ...	75	\@glxtr@noidx@do: new	167
\@Gls@: added \@glxtr@record	75	\@glxtr@redef@forglsentries: new .	6
\@GLspl@: added \@glxtr@record ...	75	\@glxtr@shortcutsval: new	23
\@gls@: added \@glxtr@record	74	\@glxtr@unsrt@getgrouptitle: new	165
\@gls@link@: added \@glxtr@record	76	\@print@noidx@glossary: added redefinition	145
\@gls@field@link: added \@glxtr@record	74	\glxtr@addloclistfield: added group key	14
\@gls@saveentrycounter: new	31	added location key	14
\@glsdisp: added \@glxtr@record ..	75	\glxtr@fields: new	154
\@glspl@: added \@glxtr@record ...	74	\glxtr@linkprefix: new	155
\@glxtr@dorecord: new	10	\glxtr@org@newignoredglossary: new	50
\@glxtr@err@undefaction: new	6	\glxtr@s@newignoredglossary: new	51
\@glxtr@record: new	7	\glxtr@shortcutsval: new	155
\@glxtr@warn@onexistsordo: new ...	6	\glxtr@texencoding: new	154
\@glxtr@warn@undefaction: new	6	\glxtr@writefields: new	155
\@print@unsrt@glossary: new	160	\GlsXtrLoadResources: new	154
record: added record package option ...	16	\glxtrpageref: new	47
\glsadd: added \@glxtr@record	81	\glxtrresourcefile: changed extension to .glstex	153
\glsdoifexists: now defines \glslabel	53	\newignoredglossary: added starred version	50
\glxtr@@do@wrglossary: new	31	1.12 (2017-02-03)	
\glxtr@addloclistfield: new	13	\@@glxtr@recordcounter: new	13
\glxtr@indexonly@saveentrycounter: new	13	\@gls@preglossaryhook: check for definition	217
\glxtr@record: new	157	\@glxtr@counterrecordhook: new .	157
\glxtr@resource: new	154	\@glxtr@display@loc: new	146
\glxtr@saveentrycounter: new	31	\@glxtr@docounterrecord: new ...	157
\glxtr@setup@record: new	13		

\@glxstr@longnewglossaryentry: new	49	\glxstrfieldifinlist: new	38
\@glxstr@noop@recordcounter: new ..	13	\glxstrfieldlistadd: new	37
\@glxstr@op@recordcounter: new ...	13	\glxstrfieldlistadd: new	37
\@glxstr@provide@storagekey: new ..	34	\glxstrfieldlistgadd: new	37
\@glxstr@s@longnewglossaryentry: new	49	\glxstrfieldlistxadd: new	37
\@glxstr@entryfmt: new	36	\glxstrfieldxifinlist: new	38
\@glxstr@indexaliased: new	99	\glxstrfmt: new	35
\@glxstr@setaliasnoindex: new	98	\GlsXtrFmtDefaultOptions: new	35
\@newglossaryentryposthook: added check for alias key	61	\GlsXtrFmtField: new	35
\@no@glxstr@indexaliased: new	99	\glxstrifkeydefined: new	34
\@printunsrtglossary: new	160	\glxstrindexaliased: new	99
General: added target key to printgloss family	140	\GlsXtrLetField: new	42
\apptoglossarypreamble: new	47	\GlsXtrLetFieldToField: new	42
\csGlsXtrLetField: new	42	\GlsXtrLoadResources: removed restriction on only one per document	154
\eGlsXtrSetField: new	43	\glxstrlocrangefmt: new	147
\gGlsXtrSetField: new	43	\glxstrpostlongdescription: new ..	50
\glxnoidxdisplayloc: added redefinition	145	\glxstrprovidestoragekey: new	34
\glssettocitle: added patch	51	\GlsXtrRecordCounter: new	157
\glxstr@counterrecord: new	157	\glxstrresourcecount: new	154
\glxstr@langtag: new	154	\glxstrresourcefile: added catcode change for @	153
\glxstr@newabbreviation: new	224	\glxstrsetaliasnoindex: new	98
\glxstr@org@newignoredglossary: Added check for existence	50	\GlsXtrSetField: new	42
\glxstr@pluralsuffixes: new	154	\glxstrsetfieldifexists: new	42
\glxstr@provideignoredglossary: new	52	\glxstrunsrtdo: new	166
\glxstr@s@newignoredglossary: Added check for existence	51	\GlsXtrusefield: new	41
\glxstr@s@provideignoredglossary: new	52	\glxstrusefield: new	41
\glxstr@abbrvpluralsuffix: new ...	229	short-postlong-user: new	329
\glxstr@alias: new	61	short-postlong-user-desc: new ...	331
\glxstr@copytoglossary: new	53	\longnewglossaryentry: added starred version	49
\glxstr@deffield: new	42	long-postshort-user: new	326
\glxstr@displayendloc: new	146	long-postshort-user-desc: new ...	328
\glxstr@displayendlochook: new ...	147	postdot: new	20
\glxstr@displayingleloc: new	146	\pretoglossarypreamble: new	48
\glxstr@displaystartloc: new	146	\print@noop@unsrtglossaryunit: new	165
\glxstr@dohyperlink: added check for alias field	103	\print@op@unsrtglossaryunit: new	165
\glxstr@deffield: new	42	\printunsrtglossary: added starred form	159
\glxstr@entryfmt: new	36	\printunsrtglossaryhandler: new ..	164
\glxstr@fieldddolistloop: new	37	\printunsrtglossaryunit: new	13
\glxstr@fieldforlistloop: new	37	\printunsrtglossaryunitsetup: new	165
		\provideignoredglossary: new	51
		\s@glxstr@provide@storagekey: new	34
		\s@printunsrtglossary: new	160
		\xGlsXtrSetField: new	43

1.13 (2017-02-07)		long-em-short-em: fixed spelling of	
\@glsdisp: removed		\glsabbrvfont	302
\@glsxtr@org@glsdisp	75	long-postshort-user: fixed spelling of	
\glsxtrsetaliasnoindex: switched to		\glsabbrvfont	326
\providecommand	98	long-postshort-user-desc: fixed	
1.14 (2017-04-18)		spelling of \glsabbrvfont	328
\@gls@link: added redefinition	78	long-short: fixed spelling of	
\@gls@noidx@getgrouptitle: new	143	\glsabbrvfont	247
\@gls@removespaces: new	147	long-short-user: fixed spelling of	
\@glsxtr@do@automake@err: new	156	\glsabbrvfont	325
\@glsxtr@org@gloautosee: new	30	footnote: fixed spelling of	
\@glsxtr@record: added third arg	7	\glsabbrvfont	252
\@glsxtr@recordsee: new	13	postfootnote: fixed spelling of	
General: added \glsadd option		\glsabbrvfont	255
theHvalue	81	1.16 (2017-06-15)	
added \glsadd option thevalue	81	\@glo@autosee: added redefinition	30
\glsdisablehyper: added redefinition	103	\@gls@noidx@getgrouptitle: fixed	
\glsenableentrycount: fixed		bug	143
assignment of \@cGls@	117	\@glsxtr@addunusedxrefs: added	
\glsenableentryunitcount: fixed		check for seealso field	62
assignment of \@cGls@	126	\@glsxtr@checkgroup: use \csuse	
\glsnavigation: new	144	instead of \csname	166
\glsxtr@org@getgrouptitle: new	144	\@glsxtr@dorecordnodefer: new	11
\glsxtr@recordsee: new	7	\@glsxtr@record@only@setup: added	
\glsxtr@writefields: added check for		check for \@gls@setupsort@none	16
automake	156	\@print@unsrt@glossary: corrected	
\glsxtrdisplayendloc: added check		misspelt command	160
for empty format	146	\@printunsrt@glossary@handler:	
\glsxtrgetgrouptitle: new	144	new	164
\glsxtrinitwrgloss: new	76	\gls@checkseeallowed: added	
\glsxtrlocationhyperlink: new	147	redefinition	30
\glsxtrsetgrouptitle: new	144	\glsxtr@writefields: added	
\glsxtrsupphypernumber: new	148	\providecommand lines	155
\ifglsxtrwrglossbefore: new	76	\glsxtrautoindex: new	210
1.15 (2017-05-10)		\glsxtrautoindexassignsort: new	210
\@glsxtr@dorecord: corrected		\glsxtrautoindexentry: new	210
premature expansion of \@glslocref	10	\glsxtrindexseealso: new	59
short-em-long-em: fixed spelling of		\glsxtrseealsolabels: new	61
\glsabbrvfont	306	\glsxtrseelist: new	58
short-long: fixed spelling of		\glsxtruseseealso: new	58
\glsabbrvfont	249	\glsxtruseseealsoformat: new	58
short-long-user: fixed spelling of		\seealso: new	58
\glsabbrvfont	332	autoindex: new	19
short-postfootnote-desc: fixed		1.17 (2017-08-09)	
spelling of \glsabbrvfont	257	\@glsxtr@mark@wordseps: new	224
short-postlong-user: fixed spelling of		\@glsxtr@markwordseps: new	224
\glsabbrvfont	329	\@glsxtr@noidx@displaynumberlist:	
short-postlong-user-desc: fixed		replace hard-coded ?? with	
spelling of \glsabbrvfont	331	\glsxtrundeftag	142

\@glxstr@noidx@entrynumberlist: replace hard-coded ?? with \glxstrundeftag	143	\Glsxtrsubsequentfmt: new	243
\@glxstr@noidx@numberlistloop: replace hard-coded ?? with \glxstrundeftag	143	\glxtrsubsequentfmt: new	242
\@glxstrifhyphenstart: new	334	\Glsxtrsubsequentplfmt: new	243
General: removed some inconsistencies in the abbreviation styles	246	\glxtrsubsequentplfmt: new	243
\glssabbrvhyphenfont: new	335	\glxtrword: new	223
\glssabbrvonlyfont: new	349	\glxtrwordsep: new	223
\glssabbrvscfont: new	266	short-hyphen-long-hyphen: new ...	344
\glssabbrvsmfont: new	282	short-hyphen-long-hyphen-desc: new	345
\glssabbrvuserfont: initialised to default font	324	short-hyphen-postlong-hyphen: new	346
\glssfirstabbrvhyphenfont: new ...	335	short-hyphen-postlong-hyphen-desc: new	348
\glssfirstabbrvonlyfont: new	349	short-long-user-desc: corrected first forms	334
\glssfirstabbrvscfont: new	266	short-nolong-desc-noreg: new ...	261
\glssfirstabbrvsmfont: new	283	short-nolong-noreg: new	259
\glssfirstlonghyphenfont: new ...	335	long-em-noshort-em-desc-noreg: new	318
\glssfirstlongonlyfont: new	349	long-em-noshort-em-noreg: new ...	314
\glslonghyphenfont: new	335	long-hyphen-noshort-desc-noreg: new	337
\glslongonlyfont: new	349	long-hyphen-noshort-noreg: new ..	339
\glslonguserfont: initialised to default font	325	long-hyphen-postshort-hyphen: new	341
\glxstr@newabbreviation: added \glxstrorgshort and \glxstrorglong	224	long-hyphen-postshort-hyphen-desc: new	343
\GlsXtrDefineAcShortcuts: new ...	22	long-hyphen-short-hyphen: new ...	335
\glxstrgenabbrvfmt: added check for \ifglxstrinsertinside	240	long-hyphen-short-hyphen-desc: new	336
\glxstrhyphensuffix: new	335	long-noshort-desc-noreg: new ...	264
\glxstrifhyphenstart: new	334	long-noshort-noreg: new	265
\glxstrlonghyphen: new	340	long-only-short-only: new	350
\glxstrlonghyphennoshort: new ...	337	long-only-short-only-desc: new ..	351
\glxstrlonghyphenshort: new	334	long-short-user-desc: corrected first forms	332
\glxstrlongshortdescname: new ...	248	1.18 (2017-08-10) stylemods: changed default value to "default"	25
\glxstronlydescname: new	351	1.19 (2017-09-09) \@glxstr@defaultnumberformat: new .	7
\glxstronlydescsort: new	351	\@glxstr@dorecord: Use \@glxstr@recordloc instead of \@glxstr@recordloc	10
\glxstronlysuffix: new	349	\@glxstr@dorecordnodefer: Use \theglentrycounter for the location rather than \@glxstr@recordloc ..	11
\glxstrtparen: new	227	\@glxstr@record@setting: new	14
\glxstrposthyphenlong: new	346	\@glxstr@record@setting@alsoindex: new	14
\glxstrposthyphenshort: new	340	\@glxstrifhasfield: new	39
\glxstrposthyphensequent: new	341		
\glxstrshortdescname: new	259		
\glxstrshorthyphen: new	346		
\glxstrshorthyphenlong: new	343		
\glxstrshortlongdescname: new ...	250		
\glxstrshortlongdescsort: new ...	250		

General: added \glslink option	
theHvalue	77
added \glslink option thevalue ...	77
\glstr@writefields: removed	
double-quotes around \jobname ..	156
\glstrdoautoindexname: changed	
format test	210
\glstrhyperlink: new	103
\glstrifhasfield: new	39
\GlsXtrSetDefaultNumberFormat:	
new	7
\s@glstrifhasfield: new	39
1.20 (2017-09-11)	
\@glstrhypernameprefix: new	140
\glsdohypertarget: added redefinition	141
\printunrtglossaryunitsetup:	
switched from redefining	
\glolinkprefix to	
\@glstrhypernameprefix	165
1.21 (2017-11-03)	
\@@glstr@record: added check for	
default options	9
\@@glstrwrglossmark: new	27
\@glslink: changed \let to \def	104
\@glstr@checkgroup: new	166
\@glstr@defpostpunc: new	19
\@glstr@do@record@wrglossary:	
new	8
\@glstr@dosee@alsoindex@glossary:	
new	29
\@glstr@doseeglossary: new	29
\@glstr@noidx@do: removed code	
dealing with the group	167
\@glstr@record@setting@off: new .	15
\@glstr@record@setting@only: new	14
\@glstr@rglstrigger@record: new	172
\@glstrglossentry: new	157
\@glstrnewgls: new	168
\@glstrsetaliasnoindex: changed to	
use \glstrifhasfield instead of	
\ifglshasfield	98
\@glstrwrglossmark: new	27
\@rGLS: new	174
\@rGLS@: new	174
\@rGLSpl: new	175
\@rGLSpl@: new	175
\@rGls: new	173
\@rGls@: new	174
\@rGlspl: new	174
\@rGlspl@: new	174
\@rGls@: new	173
\@rGlspl@: new	173
General: adjusted mcolalttree	458
modified index to remove hard coded	
\space	435
modified list to remove hard coded	
\space	424
moved conditional outside of	
\glsgroupskip	427–434
new	461
redefined altlistgroup to discourage	
breaks after group headings	425
redefined altlisthypergroup to	
discourage breaks after group	
headings	426
redefined alttreegroup to discourage	
breaks after group headings	453
redefined alttreehypergroup to	
discourage breaks after group	
headings	453
redefined indexgroup to discourage	
breaks after group headings	436
redefined indexhypergroup to	
discourage breaks after group	
headings	436
redefined listgroup to discourage	
breaks after group headings	425
redefined listhypergroup to	
discourage breaks after group	
headings	425
redefined mcolalttreegroup to	
discourage breaks after group	
headings	458
redefined mcolalttreehypergroup to	
discourage breaks after group	
headings	459
redefined mcolalttreespannav to	
discourage breaks after group	
headings	459
redefined mcolindexgroup to	
discourage breaks after group	
headings	454
redefined mcolindexhypergroup to	
discourage breaks after group	
headings	455

redefined mcolindexspannav to		\glstreechildprelocation: new ...	435
discourage breaks after group		\glstreeprelocation: new	435
headings	455	\glstriggerrecordformat: new	172
redefined mcoltreegroup to		\glseuseabbrvfont: new	240
discourage breaks after group		\glseuselongfont: new	240
headings	455	\glstr@do@alsoindex@wrglossary:	
redefined mcoltreehypergroup to		new	8
discourage breaks after group		\glstr@org@@do@wrglossary: new ..	31
headings	456	\glstr@org@dohyperlink: new	101
redefined mcoltreenonamegroup to		\glstr@setbookindexmark: new ...	466
discourage breaks after group		\glstrbookindexatendgroup: new ..	462
headings	457	\glstrbookindexbetween: new	462
redefined		\glstrbookindexbookmark: new ...	462
mcoltreenonamehypergroup to		\glstrbookindexcols: new	461
discourage breaks after group		\glstrbookindexcolspread: new ..	463
headings	457	\glstrbookindexfirstmark: new ..	467
redefined mcoltreenonamespannav to		\glstrbookindexfirstmarkfmt: new	467
discourage breaks after group		\glstrbookindexformatheader: new	462
headings	457	\glstrbookindexgroupskip: new ..	462
redefined mcoltreespannav to		\glstrbookindexlastmark: new ...	467
discourage breaks after group		\glstrbookindexlastmarkfmt: new	467
headings	456	\glstrbookindexmarkentry: new ..	466
redefined treegroup to discourage		\glstrbookindexname: new	461
breaks after group headings	439	\glstrbookindexparentchildsep:	
redefined treehypergroup to		new	462
discourage breaks after group		\glstrbookindexparentschildsep:	
headings	439	new	462
redefined treenonamegroup to		\glstrbookindexprelocation: new	461
discourage breaks after group		\glstrbookindexsubatendgroup:	
headings	441	new	462
redefined treenonamehypergroup to		\glstrbookindexsubbetween: new ..	462
discourage breaks after group		\glstrbookindexsubname: new	461
headings	441	\glstrbookindexsubprelocation:	
debug: new	28	new	461
\glssetwidest: new	443	\glstrbookindexsubsubatendgroup:	
\glsdisablehyper: added check for		new	462
existence	103	\glstrbookindexsubsubbetween:	
changed to use \def rather than \let	103	new	462
\glsenablehyper: changed to use \def		\glstrbookindexthepage: new	466
rather than \let	104	\glstrdetoklocation: new	171
\Glsfmtname: new	368	\glxtrenablerecordcount: new ...	171
\Glsfmtname: new	367	\glstrglossentry: new	157
\glshex: new	377	\glstrgroupfield: new	166
\glslstchildpostlocation: new ..	424	\Glsxrheadname: new	358
\glslstchildprelocation: new ...	424	\glxrheadname: new	358
\glslstprelocation: new	424	\GlsXtrIfFieldEqStr: new	43
\glsnahyperlink: patched	101	\glstriflabelinlist: new	165
\glssseeitemformat: new	55	\glstrifrecordtrigger: new	171
\glsshowtarget: new	28		

\glxtrindexseealso: added check that the entry exists	59	\rGLSpl: new	175
\glxtrinithyperoutside: new	77	\rGlspl: new	174
\GlsXtrLocationRecordCount: new ..	171	\rglspl: new	173
\glxtrnewgls: new	168, 169	\rGLSplformat: new	176
\glxtrnewGLSlike: new	170	\rGlsplformat: new	175
\glxtrnewglslike: new	169	\rglsplformat: new	175
\glxtrnewrgls: new	170	\s@glxtrifhasfield: switched from \ifdef to \ifundef	39
\glxtrnewrglslike: new	170	1.22 (2017-11-08)	
\glxtrprelocation: new	423, 461	\@glxtr@nopostpunc: new	139
\GlsXtrRecordCount: new	170	\@glxtr@orgprintglossary: changed explicit \let for \nopostdesc to \glxtractivatenopost	138
\glxtrrecordtriggervalue: new ..	171	\@glxtrglossentryother: new	159
\glxtrresourcefile: now disables record key	153	\glossentrynameother: new	208
\glxtrresourceinit: new	154	\glseeitemformat: switched check from regular to short	55
\GlsXtrSetRecordCountAttribute: new	171	\glxtr@setaccessdisplay: new ...	207
\Glsxtrtitlename: new	358	\glxtr@writefields: provide \glxtr@record in aux file	155
\glxtrtitlename: new	358	\glxtractivatenopost: new	138
\glxtrtitleorpdforheading: new ..	354	\glxtrbookindexprelocation: removed check for no post dot	461
\GlsXtrTotalRecordCount: new	170	\glxtrglossentryother: new	158
\glxtrwrglossmark: new	27	\glxtrnopostpunc: new	139
short-em: new	308	1.23 (2017-11-12)	
short-sc: corrected first letter uppercasing	271	\@@glxtrfmt: added check for indexing added grouping	36 35
short-sm: corrected first letter uppercasing	287	new	35
shortcuts: ac	24	\@glxtr@nopostpunc@postdesc: new	139
\ifglxtr@hyperoutside: new	77	\@glxtr@restore@postpunc: new ..	139
all: new	422	\@glxtrentryfmt: fixed missing label argument	36
nolong-short: new	261	\@glxtrfmt: new	35
nolong-short-em: new	310	\eglsupdatewidest: new	443
nolong-short-noreg: new	262	\gglupdatewidest: new	443
nolong-short-sc: new	273	\glsupdatewidest: new	443
nolong-short-sm: new	290	\GlsXtrDefineAbbreviationShortcuts: changed \newabbr definition to use \providecommand	22
nopostdot: new	20	\GlsXtrDefineAcShortcuts: changed \newabbr definition to use \providecommand	22
postpunc: new	20	\glxtrfmtdisplay: new	36
\printunsrtglossaryentryprocesshook: new	164	\glxtrifcustomdiscardperiod: new	218
\printunsrtglossarypredoglossary: new	164	\GlsXtrIfFieldUndef: new	41
\printunsrtglossaryskipentry: new	164	\glxtrrestorepostpunc: new	139
\rGLS: new	174	\s@glxtrfmt: new	35
\rGls: new	173	\s@glxtrfmt: new	35
\rgls: new	173		
\rGLSformat: new	176		
\rGlsformat: new	175		
\rglsformat: new	175		

\xglupdatewidest:new	444	\glxtrcombiningdiacriticIrules:	
1.24 (2017-11-14)		new	393
\gl sadd: added \@gls@setsort	81	\glxtrcombiningdiacriticIVrules:	
\glxtrforcsvfield:new	38	new	394
\glxtrlocalsetgrouptitle:new ..	144	\glxtrcombiningdiacriticrules:	
1.25 (2017-11-14)		new	393
\glxtrbookindexmulticol env:new	463	\glxtrcontrolrules:new	392
1.25 (2017-11-24)		\glxtrcurrencyrules:new	396
\glsextrapostnamehook:new	207	\glxtrdigitrules:new	419
\glxtrfootnotename:new	251	\glxtrfractionrules:new	420
\glxtrlongnoshortdescname:new ..	262	\glxtrGeneralLatinIIIrules:new	398
\glxtrlongnoshortname:new	265	\glxtrGeneralLatinIIrules:new ..	398
\glxtrlongshortname:new	246	\glxtrGeneralLatinIrules:new ..	397
\glxtrlongshortuserdescname:new	328	\glxtrGeneralLatinIVrules:new ..	399
\glxtronlyname:new	349	\glxtrGeneralLatinVIIrules:new	402
\glxtrpostlinkAddDescOnFirstUse:		\glxtrGeneralLatinVIrules:new ..	401
changed to use \glxtrparen	219	\glxtrGeneralLatinVrules:new ..	400
\glxtrpostlinkAddSymbolOnFirstUse:		\glxtrgeneralpuncIIrules:new ..	397
changed to use \glxtrparen	219	\glxtrgeneralpuncIrules:new ...	395
\glxtrshortlongname:new	249	\glxtrgeneralpuncrules:new	395
\glxtrshortlonguserdescname:new	330	\glxtrhyphenrules:new	395
\glxtrshortnolongname:new	257	\glxtrLatinA:new	403
1.26 (2018-01-05)		\glxtrLatinAA:new	405
\@glxtr@do@inc@linkcount:new ..	176	\glxtrLatinAELigature:new	405
\glslinkpresetkeys:new	77	\glxtrLatinE:new	403
\glxtr@inc@linkcount:new	77	\glxtrLatinEszettSs:new	404
\GlsXtrEnableLinkCounting:new ..	177	\glxtrLatinEszettSz:new	404
\GlsXtrIfLinkCounterDef:new	177	\glxtrLatinEth:new	404
\glxtrinclinkcounter:new	177	\glxtrLatinH:new	403
\GlsXtrLinkCounterName:new	177	\glxtrLatinI:new	403
\GlsXtrLinkCounterValue:new	177	\glxtrLatinInsularG:new	405
\GlsXtrTheLinkCounter:new	177	\glxtrLatinK:new	403
1.27 (2018-02-26)		\glxtrLatinL:new	403
\@glxtrdialecthook:new	32	\glxtrLatinLslash:new	405
General: added		\glxtrLatinM:new	403
glossaries-extra-bib2gls.sty	376	\glxtrLatinN:new	403
\Alpha:new	389	\glxtrLatinO:new	403
\Beta:new	389	\glxtrLatinOELigature:new	405
\Chi:new	390	\glxtrLatinOslash:new	405
\Digamma:new	390	\glxtrLatinP:new	404
\Epsilon:new	389	\glxtrLatinS:new	404
\Eta:new	389	\glxtrLatinSchwa:new	404
\glxtr@loaddialect:new	376	\glxtrLatinT:new	404
\glxtrBasicDigitrules:new	419	\glxtrLatinThorn:new	404
\glxtrcombiningdiacriticIIrules:		\glxtrLatinWynn:new	405
new	394	\glxtrLatinX:new	404
\glxtrcombiningdiacriticIrules:		\glxtrMathGreekIIrules:new	411
new	393	\glxtrMathGreekIrules:new	410

\glstredfield: changed \csedef to \protected\csedef 42	\glsaddpresetkeys: new 81
\glsxtrlocalsetgrouptitle: changed \csedef \protected\csedef 144	\glsuserdescription: new 325
\glsxtrsetgrouptitle: changed \csxdef \protected\csxdef 144	\glsxtrabbreviationfont: new 74
1.29 (2018-04-09)	\GlsXtrDualBackLink: new 379
\@gls@removespaces: added expansion 147	\GlsXtrDualField: new 379
\@glsxtr@dorecord: don't suppress expansion of \@glsrecordlocref if counter isn't page 11	\GlsXtrExpandedFmt: new 77
\@glsxtr@wrglossary@locationhyperlink: new 27	\GLSxtrlong: added \@glsxtr@record 236
\glsxtr@inc@wrglossaryctr: new ... 26	\Glsxtrlong: added \@glsxtr@record 235
\glsxtr@wrglossarylocation: new . 378	\glsxtrlong: added \@glsxtr@record 235
\GlsXtrBibTeXEntryAliases: new .. 379	\GLSxtrlongpl: added \@glsxtr@record 240
\glsxtrfieldforlistloop: corrected argument order in \forlistcsloop 37	\Glsxtrlongpl: added \@glsxtr@record 239
\GlsXtrIndexCounterLink: new 378	\glsxtrlongpl: added \@glsxtr@record 239
\GlsXtrInternalLocationHyperlink: new 26	\GLSxtrshort: added \@glsxtr@record 234
\GlsXtrProvideBibTeXFields: new . 379	\Glsxtrshort: added \@glsxtr@record 234
indexcounter: new 27	\glsxtrshort: added \@glsxtr@record 233
\setentrycounter: new 147	\GLSxtrshortpl: added \@glsxtr@record 238
1.30 (2018-04-25)	\Glsxtrshortpl: added \@glsxtr@record 237
\@glsxtr@record: added check for post-key hook 9	\glsxtrshortpl: added \@glsxtr@record 237
added check for pre-key hook 9	\GlsXtrStartUnsetBuffering: new . 113
\@GLSxtr@fullpl: added \@glsxtr@record 232	\GlsXtrStopUnsetBuffering: new .. 114
\@GlsXtrStopUnsetBuffering: new . 114	indexcounter: added check for wrglossary counter 27
\@Glsxtr@fullpl: added \@glsxtr@record 232	\s@GlsXtrStopUnsetBuffering: new 114
\@glsxtr@dorecord: don't suppress expansion of \@glsrecordlocref .. 11	1.31 (2018-05-09)
\@glsxtr@full: added \@glsxtr@record 230	\@GlsXtrStartUnsetBuffering: new 113
\@glsxtr@fullpl: added \@glsxtr@record 231	\@gls@ifaccessattribute@set: new 186
\@glsxtr@glossadd@postkeys: new .. 10	\@gls@initaccesskeys: new ... 186, 194
\@glsxtr@glossadd@prekeys: new ... 10	\@gls@setup@default@short@access: new 188
\@glsxtr@glslink@postkeys: new ... 10	\@glsxtr@record@noglossarywarning: new 152
\@glsxtr@glslink@prekeys: new 10	\@glsxtrbuffer@nodup@unset: new . 114
\@glsxtr@local@textformat: new ... 77	General: added prefix key for glslink . 77
\@glsxtr@unset: new 113	added prefix key for printgloss .. 140
\@glsxtrbuffer@unset: new 114	changed \let to \def 140
\glsadd: added \glsaddpostsetkeys . 81	\glsaddeach: new 82
added \glsaddpresetkeys 81	\glsapturedgroup: new 377
\glsaddpostsetkeys: new 81	\glsdefpostdesc: new 218
	\glsdefpostlink: new 218
	\glsdefpostname: new 207

<code>\glsdohypertarget</code> : bug fix: ensure that new version is picked up	141	<code>\if@glxtrdocdefrestricted</code> : changed to allow for atom as well ...	18
<code>\glslistdesc</code> : new	424	<code>docdef</code> : atom	18
<code>\glslocalreseteach</code> : new	115	1.35 (2018-08-13)	
<code>\glslocalunseteach</code> : new	115	<code>\@gls@link@</code> : initialise post-link hook commands	76
<code>\glstreechilddesc</code> : new	438	1.36 (2018-08-18)	
<code>\glstreechildsymbol</code> : new	438	<code>\glxtrautoindexesc</code> : new	210
<code>\glstreedefaultnamefmt</code> : new	434	<code>\glxtrdisplaysuploc</code> : new	380
<code>\glstreedesc</code> : new	437	<code>\glxtrmultisupplocation</code> : new ...	380
<code>\glstreegroupheaderfmt</code> : added redefinition	435	1.37 (2018-11-30)	
<code>\glstreenamefmt</code> : added redefinition .	435	<code>\@glxtr@record</code> : added check for auto-add	9
<code>\glstreenavigationfmt</code> : added redefinition	435	<code>\@dGLS</code> : new	388
<code>\glstreenonamechilddesc</code> : new	440	<code>\@dGLSpl</code> : new	389
<code>\glstreenonamedesc</code> : new	439	<code>\@dGls</code> : new	388
<code>\glstreenonamesymbol</code> : new	440	<code>\@dGlspl</code> : new	388
<code>\glstreesymbol</code> : new	438	<code>\@dgls</code> : new	388
<code>\glxtr@newabbreviation</code> : added <code>\ExtraCustomAbbreviationFields</code>	224	<code>\@dglspl</code> : new	388
<code>\GlsXtrForUnsetBufferedList</code> : new	114	<code>\@gls@getcounterprefix</code> : new	31
<code>\GlsXtrIfFieldCmpNum</code> : new	40	<code>\@gls@longextrawidestname</code> : new ...	470
<code>\GlsXtrIfFieldEqNum</code> : new	40	<code>\@glxtr@bibgls@removespaces</code> : new	382
<code>\GlsXtrIfFieldEqXpStr</code> : new	43	<code>\@glxtr@check@bibgls@nameref</code> : new	154
<code>\GlsXtrIfFieldNonZero</code> : new	39	<code>\@glxtr@do@nameref@record</code> : new ..	12
<code>\GlsXtrIfHasNonZeroChildCount</code> : new	378	<code>\@glxtr@get@prefixedlabel</code> : new .	387
<code>\GlsXtrIfXpFieldEqXpStr</code> : new	44	<code>\@glxtr@if@record@only</code> : new	15
<code>\glxtrpostlinkAddSymbolDescOnFirstUse</code> : new	219	<code>\@glxtr@ifnum@mmode</code> : new	12
<code>\GlsXtrRecordWarning</code> : new	151	<code>\@glxtr@labelprefixes</code> : new	386
<code>\glxtrRevertToCmarks</code> : new	353	<code>\@glxtr@prefixlabellist</code> : new ...	387
<code>\GlsXtrStandaloneGlossaryType</code> : new	158	<code>\@glxtr@providenewgls</code> : new	168
<code>\GlsXtrStandaloneSubEntryItem</code> : new	158	<code>\@glxtr@record@only@setup</code> : new ..	15
<code>\s@GlsXtrStartUnsetBuffering</code> : new	113	<code>\@glxtr@record@setting@nameref</code> : new	15
1.32 (2018-05-24)		<code>\@glxtr@use@equation@counter@or</code> : new	77
<code>\GlsXtrForeignText</code> : new	45	General: new	468
<code>\GlsXtrForeignTextField</code> : new	47	page: nameref	11
<code>\GlsXtrUnknownDialectWarning</code> : new	47	<code>\dGLS</code> : new	388
1.33 (2018-07-26)		<code>\dgls</code> : new	387
<code>\ifglsused</code> : added redefinition	48	<code>\dglsdisp</code> : new	389
1.34 (2018-07-29)		<code>\dglslink</code> : new	389
<code>\gls@begindocdefs</code> : atom	63	<code>\dGLSpl</code> : new	388
<code>\GlsXtrIfUnusedOrUndefined</code> : new ..	32	<code>\glsadd</code> : added grouping	81
<code>\glxtrNoGlossaryWarning</code> : added package warning	25	ensure that <code>\glsadd</code> performs indexing	81
		<code>\gls@longextraDescAlign</code> : new	470
		<code>\gls@longextraDescFmt</code> : new	468
		<code>\gls@longextraDescNameHeader</code> : new	475

\glslongextraDescNameTabularFooter:	\glslongextraNameDescSymTabularFooter:
new 475	new 478
\glslongextraDescNameTabularHeader:	\glslongextraNameDescSymTabularHeader:
new 475	new 478
\glslongextraDescSymNameHeader:	\glslongextraNameDescTabularFooter:
new 487	new 470
\glslongextraDescSymNameTabularFooter:	\glslongextraNameDescTabularHeader:
new 487	new 470
\glslongextraDescSymNameTabularHeader:	\glslongextraNameFmt: new 468
new 487	\glslongextraNameSymDescHeader:
\glslongextraGroupHeading: new .. 470	new 481
\glslongextraHeaderFormat: new .. 470	\glslongextraNameSymDescLocationHeader:
\glslongextraLocationAlign: new . 470	new 482
\glslongextraLocationDescNameHeader:	\glslongextraNameSymDescLocationTabularFooter:
new 476	new 483
\glslongextraLocationDescNameTabularFooter:	\glslongextraNameSymDescLocationTabularHeader:
new 476	new 482
\glslongextraLocationDescNameTabularHeader:	\glslongextraNameSymDescTabularFooter:
new 476	new 481
\glslongextraLocationDescSymNameHeader:	\glslongextraNameSymDescTabularHeader:
new 488	new 481
\glslongextraLocationDescSymNameTabularFooter:	\glslongextraSetDescWidth: new .. 471
new 489	\glslongextraSetWidest: new 470
\glslongextraLocationDescSymNameTabularHeader:	\glslongextraSubDescFmt: new 469
new 489	\glslongextraSubLocationFmt: new 469
\glslongextraLocationFmt: new ... 469	\glslongextraSubNameFmt: new 469
\glslongextraLocationSymDescNameHeader:	\glslongextraSubSymbolFmt: new .. 469
new 485	\glslongextraSymbolAlign: new ... 470
\glslongextraLocationSymDescNameTabularFooter:	\glslongextraSymbolFmt: new 468
new 486	\glslongextraSymDescNameHeader:
\glslongextraLocationSymDescNameTabularHeader:	new 484
new 485	\glslongextraSymDescNameTabularFooter:
\glslongextraLocSetDescWidth: new 472	new 484
\glslongextraNameAlign: new 470	\glslongextraSymDescNameTabularHeader:
\glslongextraNameDescHeader: new 470	new 484
\glslongextraNameDescLocationHeader:	\glslongextraSymLocSetDescWidth:
new 473	new 472
\glslongextraNameDescLocationTabularFooter:	\glslongextraSymSetDescWidth: new 471
new 474	\glslongextraTabularVAlign: new . 472
\glslongextraNameDescLocationTabularHeader:	\glslongextraUpdateWidest: new .. 471
new 473	\glslongextraUpdateWidestChild:
\glslongextraNameDescSymHeader:	new 471
new 478	\glsrenewcommand: new 378
\glslongextraNameDescSymLocationHeader:	\glsseeitemformat: removed reference
new 479	to \glslabel 55
\glslongextraNameDescSymLocationTabularFooter:	\glsxtr@dblfloat: new 19
new 479	\glsxtr@do@autoadd: new 78
\glslongextraNameDescSymLocationTabularHeader:	\glsxtr@float: new 19
new 479	\glsxtr@record@nameref: new 157

\glstr@renewcommand: new	378	1.38 (2018-12-01)	
\glstr@writefields: provide			\glslongextraNameFmt: bug fix:
\glstr@record@nameref in aux			removed double param
file	155		all: added glossary-longextra
\glstraddlabelprefix: new	386	1.39 (2019-03-22)	
\GlsXtrAutoAddOnFormat: new	78		\@GlsXtrIfFieldCmpNum: new
\glstrclearlabelprefixes: new	386		\@GlsXtrIfFieldEqNum: new
\glstrdisplaylocnameref: new	380		\@GlsXtrIfFieldEqStr: new
\glstrfmtexternalnameref: new	383		\@GlsXtrIfFieldEqXpStr: new
\glstrfmtinternalnameref: new	383		\@GlsXtrIfFieldNonZero: new
\GLSXRhiername: new	57		\@GlsXtrIfXpFieldEqXpStr: new
\GLSxtrhiername: new	57		\@glstr@removespaces: changed \x to
\GlsXtrhiername: new	56		\@glo@tmp
\Glsxtrhiername: new	56		\@glstr@dorecord: added protection
\glstrhiername: new	56		for fragile commands
\glstrhiernamesep: new	58		General: added label key for
\glstridentifyglslike: new	168		printgloss
\glstrifinlabelprefixlist: new	387		\glstrbookindexlocation: new
\GlsXtrLocationField: new	166		\glstrbookindexsublocation: new
\glstrnameloclink: new	382		\glstrentryparentname: new
\glstrnamerefink: new	381		\GlsXtrIfFieldCmpNum: added starred
\glstrprependlabelprefix: new	386		version
\GlsXtrSetAltModifier: write modifier			\GlsXtrIfFieldEqNum: added starred
to aux	101		version
\glstrSetWidest: new	383		\GlsXtrIfFieldEqStr: added starred
\glstrSetWidestFallback: new	385		form
\GlsXtrStandaloneEntryName: new	158		\GlsXtrIfFieldEqXpStr: added starred
\GlsXtrStandaloneEntryOther: new	159		form
\GLSxtrusefield: new	41		\GlsXtrIfFieldNonZero: added starred
\Glsxtrusefield: fixed internal			version
command and added check for			\GlsXtrIfXpFieldEqXpStr: added
\texorpdfstring	41		starred form
\ifGlsLongExtraUseTabular: new	472		\glstrsetglossarylabel: new
floats: new	19		\glstrshortdescname: corrected to
long-desc-name: new	475		show long form as advertised in the
long-desc-sym-name: new	487		manual
long-loc-desc-name: new	477		short-desc: corrected to omit
long-loc-desc-sym-name: new	489		description key as advertised in the
long-loc-sym-desc-name: new	486		manual
long-name-desc: new	472		short-em-desc: bug fix: omit description
long-name-desc-loc: new	474		key as advertised in the manual
long-name-desc-sym: new	478		short-sc-desc: bug fix: omit description
long-name-desc-sym-loc: new	480		key as advertised in the manual
long-name-sym-desc: new	481		short-sm-desc: corrected to omit
long-name-sym-desc-loc: new	483		description key as advertised in the
long-sym-desc-name: new	484		manual
equations: new	19		\s@GlsXtrIfFieldCmpNum: new
			\s@GlsXtrIfFieldEqNum: new
			\s@GlsXtrIfFieldEqStr: new

\s@GlsXtrIfFieldEqXpStr: new	44	1.41 (2019-04-09)	General: changed \thisgrptitle to	
\s@GlsXtrIfXpFieldEqXpStr: new . . .	44		\glxtrcurrentgrptitle	466
1.4.2 (??)			\glslistgroupskip: new	424
\@glossentrysymbol: new	214		\glstopicAssignSubIndent: moved	
\glstentrypdfsymbol: new	214		\par from \glstopicSubItem	493
1.40 (2019-03-22)			\glstopicSubItem: added check for	
General: new	491		description	494
\glstopicAssignSubIndent: new . . .	493		moved \par to	
\glstopicAssignWidest: new	494		\glstopicAssignSubIndent	494
\glstopicCols: new	495		\glstopicSubLoc: moved \space to	
\glstopicColsEnv: new	495		\glstopicSubPreLocSep	495
\glstopicDesc: new	493		\glstopicSubPreLocSep: new	495
\glstopicGroupHeading: new	492		\glstreeChildDescLoc: new	438
\glstopicInit: new	493		\glstreeDescLoc: new	437
\glstopicItem: new	492		\glstreegroupskip: new	435
\glstopicLoc: new	493		\glstreePreHeader: new	435
\glstopicMarker: new	492		\glxtralttreeSymbolDescLocation:	
\glstopicMidSkip: new	494		added check for description	442
\glstopicName: new	492		topic: added penalty if no description .	491
\glstopicParIndent: new	493		topicmcols: added penalty if no	
\glstopicPostSkip: new	494		description	496
\glstopicPreSkip: new	494	1.42 (2020-02-03)	\@@glxtr@record: moved label	
\glstopicSubIndent: new	493		definition outside of conditional	9
\glstopicSubItem: new	494		\@ACRfull: added redefinition	94
\glstopicSubItemBox: new	495		\@ACRfullpl: added redefinition	95
\glstopicSubItemSep: new	495		\@Acrfull: added redefinition	94
\glstopicSubLoc: new	495		\@Acrfullpl: added redefinition	95
\glstopicSubNameFont: new	495		\@GlsXtrIfFieldValueInCsvList:	
\glstopicTitleFont: new	493		new	38
\glstopicwidest: new	493		\@acrfull: added redefinition	94
all: added glossary-topic	422		\@acrfullpl: added redefinition	95
topic: new	491		\@gl@assign@actual: new	187
topicmcols: new	495		\@gl@entry@field: redefined	48
1.40 (2019-03-31)			\@gl@setup@default@access: added	
\glfirstabbrvdefaultfont: changed			\gl@defaultshortaccess	188
definition from \gl@abbrvfont to			\@gl@setup@default@short@access:	
\gl@abbrvdefaultfont for			renamed to	
consistency	229		\@gl@setup@default@access . .	188
\GlsXtrDefaultResourceOptions:			\@gl@link: switched from	
new	153		\gl@dohyperlink to	
long-hyphen-noshort-noreg:			\glxtr@dohyperlink	104
corrected formatting commands . . .	340		\@glxtr@abbrlists: new	130
\printunsrtabbreviations: new . . .	377		\@glxtr@acronymlists: new	130
\printunsrtacronyms: new	376		\@glxtr@addabbreviationlist: new	130
\printunsrtindex: new	376		\@glxtr@base@acrcmd: new	90
\printunsrtnumbers: new	377		\@glxtr@doloadprefix: new	24
\printunsrtsymbols: new	377			

\@glstr@org@addtoacronymlists: new	130	removed \relax and updated \@glshort instead of \glsshorttok	225
\@glstr@org@setacronymlists: new	130	replaced explicit \spacefactor with \@	225
\@glstrentryfmt: added \glslabel and scope	36	\glstr@writefields: added check for order=letter	156
General: added \@afterheading	455	\glstrAccSuppAbbrSetFirstLongAttrs: new	190, 195
debug: showaccsupp	28	\glstrAccSuppAbbrSetNameLongAttrs: new	191, 195
\forall abbreviationlists: new ...	130	\glstrAccSuppAbbrSetNameShortAttrs: new	191, 195
\forall acronymlists: new	130	\glstrAccSuppAbbrSetNoLongAttrs: new	190, 194
\gldefaultshortaccess: new	186	\glstrAccSuppAbbrSetTextShortAttrs: new	191, 195
\gldisplaynumberlist: added	377	\glstralttreeSymbolDescLocation: switched to using \glstreeDescLoc	442
\glenablehyper: switched from \glsdohyperlink to \glstrdohyperlink	104	\glstrassignactualsetup: new ...	187
\glentrynumberlist: added	377	\glstrbookindexbookmarkprefix: new	463
\GLSfmtfirst: new	370	\GlsXtrDiscardUnsetBuffering: new	114
\GLSfmtfirstpl: new	371	\glstrdohyperlink: new (was former redefinition of \glsdohyperlink) .	102
\GLSfmtfull: new	374	\glstrequationlocfmt: new	381
\Glsfmtfull: switched pdf case to use \glspdffmtfull	374	\glstrfieldformatcsvlist: new ...	38
\glspdffmtfull: switched pdf case to use \glspdffmtfull	373	\glstrfieldformatlist: new	37
\GLSfmtfullpl: new	375	\glstrfootnotedescname: new ...	253
\Glsfmtfullpl: switched pdf case to use \glspdffmtfullpl	375	\glstrfootnotedescsort: new ...	253
\glspdffmtfullpl: switched pdf case to use \glspdffmtfullpl	374	\GLSXtrhiername: switched to using \GLSfmtttext and \GLSfmtname ...	57
\GLSfmtlong: new	372	\GLSXtrhiername: switched to using \glsfmmttext, \glsfmtname, \GLSfmtttext and \GLSfmtname ...	57
\GLSfmtlongpl: new	373	\GlsXtrhiername: switched to using \Glsfmmttext and \Glsfmtname ...	56
\GLSfmtname: new	368	\GlsXtrhiername: switched to using \glsfmmttext and \glsfmtname ...	56
\GLSfmtplural: new	370	\glstrIfFieldValueInCsvList: new	38
\GLSfmtttext: new	369	\glstrpdfentryfmt: new	36
\glspdffmtfull: new	373	\glstrprovideaccsuppcmd: new ...	190
\glspdffmtfullpl: new	373	\glstrscsuffix: added \protect ..	266
\glseeitemformat: switched to using \glsfmmttext and \glsfmtname ...	55	\GlsXtrSetAltModifier: added check	101
\glsshowtarget: added check for \glsshowtargetouter	28	\GLSxtrtitlefirst: new	361
\glstreeChildDescLoc: added \glstreeNoDescSymbolPreLocation	438	\GLSxtrtitlefirstplural: new ...	362
\glstreegroupheaderskip: new ...	435	\GLSxtrtitlefull: new	365
\glstreeNoDescSymbolPreLocation: new	437		
\glstr@newabbreviation: moved apply abbreviation style to after category key has been obtained ...	224		

\GLSxtrtitlefullpl: new	366	moved \protect inside	
\GLSxtrtitlelong: new	363	\glxtrscsuffix	278
\GLSxtrtitlelongpl: new	364	short-sc-footnote-desc: new	279
\GLSxtrtitlename: new	358	short-sc-long: added missing text key	268
\GLSxtrtitleplural: new	360	moved \protect inside	
\GLSxtrtitleshort: new	357	\glxtrscsuffix	269
\GLSxtrtitleshortpl: new	358	short-sc-postfootnote: added	
\GLSxtrtitletext: new	359	missing text key	280
\glxtrusealias: new	58	moved \protect inside	
short-em: removed \protect from		\glxtrscsuffix	280
\glxtremsuffix	308	short-sc-postfootnote-desc: new .	281
short-em-desc: removed \protect		short-sm: removed \protect from	
from \glxtremsuffix	309	\glxtrsmsuffix	287
short-em-footnote: added missing text		short-sm-desc: removed \protect	
key	318	from \glxtrsmsuffix	289
removed \protect from		short-sm-footnote: added missing text	
\glxtremsuffix	319	key	294
short-em-footnote-desc: new	320	removed \protect from	
short-em-long: added missing text key	303	\glxtrsmsuffix	294
removed \protect from		short-sm-footnote-desc: new	296
\glxtremsuffix	304	short-sm-long: added missing text key	285
short-em-long-em: added missing text		removed \protect from	
key	305	\glxtrsmsuffix	285
removed \protect from		short-sm-postfootnote: added	
\glxtremsuffix	306	missing text key	296
short-em-postfootnote: added		removed \protect from	
missing text key	321	\glxtrsmsuffix	297
removed \protect from		short-sm-postfootnote-desc: new .	298
\glxtremsuffix	322	\makeglossaries: added	
short-em-postfootnote-desc: new .	323	\@domakeglossaries	133
short-footnote-desc: new	254	let \@makeglossary to \@gobble	
short-hyphen-long-hyphen: added		instead of \relax	134
missing text key	344	removed redefinition of	
short-hyphen-postlong-hyphen:		\makeglossary	134
added missing text key	346	\makenoidxglossaries: added	
short-long: added missing text key ...	249	\@domakeglossaries	64
short-long-user: added missing text		long-em-noshort-em: removed	
key	332	\protect from \glxtremsuffix .	313
short-postfootnote-desc: added		long-em-noshort-em-desc: removed	
missing text key	257	\protect from \glxtremsuffix .	317
new	256	long-em-short-em: added missing text	
short-postlong-user: added missing		key	301
text key	329	removed \protect from	
short-sc: moved \protect inside		\glxtremsuffix	302
\glxtrscsuffix	270	long-hyphen-noshort-desc-noreg:	
short-sc-desc: moved \protect inside		added missing text key	337
\glxtrscsuffix	272	long-hyphen-postshort-hyphen:	
short-sc-footnote: added missing text		added missing text key	341
key	277		

long-hyphen-short-hyphen: added missing text key	335	\@glxstr@noidx@do: replaced \ifglshasparent with \@glxstr@ifischild	167
long-noshort-em: removed \protect from \glxstremsuffix	311	\@print@unsrt@innerglossary: new	163
long-noshort-em-desc: removed \protect from \glxstremsuffix	315	General: added groups key	141
long-noshort-sc: moved \protect inside \glxtrscsuffix	274	added leveloffset key	140
long-noshort-sc-desc: moved \protect inside \glxtrscsuffix	276	\doifglossarynoexistsordo: switched to starred form of \ifglossaryexists	54
long-noshort-sm: removed \protect from \glxtrsmsuffix	291	\glswriteentry: replaced \ifglused with \GlsXtrIfUnusedOrUndefined	99
long-noshort-sm-desc: removed \protect from \glxtrsmsuffix	292	\glxstr@printgloss@checkexists: new	137
long-only-short-only: added missing text key	350	\glxtralttreeSymbolDescLocation: removed duplicate description	442
removed \protect from \glxstronlysuffix	350	\ifglossaryexists: added check for starred form	33
long-postshort-user: added missing text key	326	\np@glxstr@assign@leveloffset: new	141
long-short: added missing text key	247	\p@glxstr@assign@leveloffset: new	141
long-short-em: added missing text key	299	\pp@glxstr@assign@leveloffset: new	141
removed \protect from \glxstremsuffix	300	\printunsrtglossary: added check for \@printgloss@checkexists	159
long-short-sc: added missing text key	266	\printunsrtinnerglossary: new	161
moved \protect inside \glxtrscsuffix	267	printunsrtglossarywrap: new	161
long-short-sm: added missing text key	283	1.45 (2020-04-01)	
removed \protect from \glxtrsmsuffix	283	General: removed duplicate description	440
long-short-user: added missing text key	325	\glstreenonameChildDescLoc: new	440
footnote: added missing text key	252	\glstreenonameDescLoc: new	440
footnote-desc: new	254	1.46 (2021-09-18)	
postfootnote: added missing text key	255	\@glxstrsetaliasnoindex: changed to use starred version of \glxstrifhasfield	98
prefix: new	25	1.46 (2021-09-20)	
\RestoreAcronyms: added display style	132	\@@glxstr@record: changed \edef to \protected@edef	9
\s@GlsXtrIfFieldValueInCsvList: new	39	\@@newglossaryentry@defunitcounters: changed \edef to \protected@edef	122
\seealso: add check for \also	58	\@glossentrysymbol: changed \edef to \protected@edef	214
1.42 (?)		\@gls@increment@currunitcount: changed \edef to \protected@edef	123
postfootnote-desc: new	257	\@gls@link: changed \edef to \protected@edef	78, 79
1.43 (2020-02-28)		\@gls@link@checkfirsthyper: changed \edef to \protected@edef	97
\@glxtrentryfmt: changed \def to \edef to avoid infinite recursion	36	\@gls@local@increment@currunitcount: changed \edef to \protected@edef	123
1.44 (2020-03-23)			
\@glxstr@assign@leveloffset: new	140		
\@glxstr@leveloffset: new	140		

<code>\@gls@setup@default@access:</code>	General: changed <code>\edef</code> to
changed <code>\edef</code> to	<code>\protected@edef</code> 61, 222
<code>\protected@edef</code> 188, 189	record: added hybrid 16
<code>\@glsxtr@addabbreviationlist:</code>	<code>\glossentrydesc:</code> changed <code>\edef</code> to
changed <code>\eappto</code> to	<code>\protected@edef</code> 201, 202
<code>\protected@eappto</code> 130	<code>\Glossentryname:</code> changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> 130	<code>\protected@edef</code> 205, 206
<code>\@glsxtr@bibgls@removespaces:</code>	<code>\glossentryname:</code> changed <code>\edef</code> to
changed <code>\x</code> to <code>\@glo@tmp</code> 382	<code>\protected@edef</code> 203, 204
<code>\@glsxtr@do@inc@linkcount:</code> changed	<code>\glossentrynameother:</code> changed <code>\edef</code>
<code>\x</code> to <code>\@glo@tmp</code> 176	to <code>\protected@edef</code> 208
<code>\@glsxtr@do@record@wrglossary:</code>	<code>\glsadd:</code> changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> . 8	<code>\protected@edef</code> 81
<code>\@glsxtr@do@redef@forglseentries:</code>	<code>\glsalttreechildpredesc:</code> new 441
changed <code>\edef</code> to <code>\protected@edef</code> . 6	<code>\glsalttreepredesc:</code> new 441
<code>\@glsxtr@get@prefixedlabel:</code>	<code>\glsdisablehyper:</code> changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> 387	<code>\protected@edef</code> 103
changed <code>\x</code> to <code>\@glo@tmp</code> 387	<code>\glsdoifexists:</code> changed <code>\edef</code> to
<code>\@glsxtr@mixed@assign@sortkey:</code>	<code>\protected@edef</code> 53
changed <code>\edef</code> to <code>\protected@edef</code> 141	<code>\glsenableentryunitcount:</code> changed
<code>\@glsxtr@op@recordcounter:</code> changed	<code>\edef</code> to <code>\protected@edef</code> ... 125, 126
<code>\eappto</code> to <code>\protected@eappto</code> ... 13	<code>\glsFindWidestLevelTwo:</code> changed
<code>\@glsxtr@orgprintglossary:</code> changed	<code>\edef</code> to <code>\protected@edef</code> 447
<code>\xdef</code> to <code>\protected@xdef</code> 138	<code>\glsFindWidestUsedLevelTwo:</code>
<code>\@glsxtr@rglstrigger@record:</code>	changed <code>\edef</code> to <code>\protected@edef</code> 446
changed <code>\edef</code> to <code>\protected@edef</code> 172	<code>\glsnavhyperlink:</code> changed <code>\edef</code> to
<code>\@glsxtr@warn@hybrid@noprintgloss:</code>	<code>\protected@edef</code> 102
new 15	<code>\glstopicAssignSubIndent:</code> bug 182
<code>\@glsxtrentryfmt:</code> changed <code>\edef</code> to	maintain hangindent for multiple
<code>\protected@edef</code> 36	paragraphs 493
<code>\@glsxtrglossentry:</code> changed <code>\edef</code> to	<code>\glstopicsubitemhangindent:</code> new . 493
<code>\protected@edef</code> 158	<code>\glstopicSubItemParIndent:</code> new .. 493
<code>\@glsxtrglossentryother:</code> changed	<code>\glsxtr@org@newignoredglossary:</code>
<code>\edef</code> to <code>\protected@edef</code> 159	changed <code>\eappto</code> to
<code>\@glsxtrindexaliased:</code> changed <code>\edef</code>	<code>\protected@eappto</code> 50
to <code>\protected@edef</code> 99	changed <code>\edef</code> to <code>\protected@edef</code> . 50
<code>\@makeglossaries@warn@noprintglossary:</code>	<code>\glsxtr@provideignoredglossary:</code>
new 133	changed <code>\eappto</code> to
<code>\@newglossaryentryposthook:</code>	<code>\protected@eappto</code> 52
changed <code>\edef</code> to	changed <code>\edef</code> to <code>\protected@edef</code> . 52
<code>\protected@edef</code> 61, 62	<code>\glsxtr@s@newignoredglossary:</code>
<code>\@print@unsrt@glossary:</code> changed	changed <code>\edef</code> to <code>\protected@edef</code> 51
<code>\eappto</code> to <code>\protected@eappto</code> .. 161	<code>\glsxtr@s@provideignoredglossary:</code>
<code>\@print@unsrt@innerglossary:</code>	changed <code>\edef</code> to <code>\protected@edef</code> 52
changed <code>\eappto</code> to	<code>\glsxtr@setaccessdisplay:</code> changed
<code>\protected@eappto</code> 164	<code>\edef</code> to <code>\protected@edef</code> 207
<code>\@printunsrt@glossary@handler:</code>	<code>\glsxtralttreeSymbolDescLocation:</code>
changed <code>\xdef</code> to <code>\protected@xdef</code> 164	switch to using <code>\glsalttreepredesc</code>
	and <code>\glsalttreechildpredesc</code> .. 442

<code>\glstrdisplayendloc</code> : changed <code>\edef</code> to <code>\protected@edef</code>	146	changed <code>\edef</code> to <code>\protected@edef</code>	134, 135
<code>\glstrdisplaystartloc</code> : changed <code>\edef</code> to <code>\protected@edef</code>	146	<code>\makenoidxglossaries</code> : changed <code>\edef</code> to <code>\protected@edef</code>	64
<code>\glstrdoautoindexname</code> : changed <code>\eappto</code> to <code>\protected@eappto</code> ..	210	topic: added <code>\par</code> (bug 176)	492
<code>\glstrseelist</code> : changed <code>\edef</code> to <code>\protected@edef</code>	58	grouping added to scope <code>\everypar</code> (bug 182)	492
<code>\glstrtreechildpredesc</code> : new	437	<code>printunsrtglossarywrap</code> : changed <code>\xdef</code> to <code>\protected@xdef</code>	162
<code>\glstrtreepredesc</code> : new	437	<code>\setabbreviationstyle</code> : changed <code>\edef</code> to <code>\protected@edef</code>	244
<code>\makeglossaries</code> : adjust warning on missing glossary for “alsoindex” ...	133		

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	
\\$	377
\,	58
\.	20, 219, 434
\@	63, 153, 187, 225
\@@cGLS@	117, 126
\@@cGLSpl@	118, 126
\@@cGls@	117, 126
\@@cGlspl@	117, 126
\@@cgls@	117, 126
\@@cglspl@	117, 126
\@do@wrglossary	8, 10, 135
\@do@wrglossary	13, 15–17, 31, 81, 99
\@glo@assign@sortkey	141
\@glo@list	6
\@glo@type	160
\@glossarysec	462
\@glossaryseclabel	140
\@gls@expand@field	34
\@glslocalreset	115
\@glslocalunset	115
\@glsreset	115
\@glsunset	113
\@glsxtr@autoindex@escspch	212, 213
\@glsxtr@base@acrcmd@warn	90, 131
\@glsxtr@checkspch	211, 213, 214
\@glsxtr@disabledflycommand	69
\@glsxtr@org@postdescription	139
\@glsxtr@record	15, 17
\@glsxtr@recordcounter	15–17, 157
\@glsxtrfmt	35
\@glsxtrp	108, 109
\@glsxtrpostloctag	71
\@glsxtrpreloctag	71, 72
\@glsxtrwrglossmark	8, 9, 13, 29, 31, 59, 64, 135
\@@newglossaryentry@defcounters	116
\@@newglossaryentry@defunitcounters	124
\@par	442
\@ACRlong	105
\@ACRlongpl	105
\@ACRshort	105
\@ACRshortpl	105
\@Acrlong	105
\@Acrlongpl	105
\@Acrshort	105
\@Acrshortpl	105
\@GLS@	104, 120, 121, 174, 388
\@GLSdesc@	86
\@GLSpl@	104, 120, 121, 175, 389
\@GLSplural@	105
\@GLSsymbol@	87
\@GLSxtr@full	231
\@GLSxtr@fullpl	232
\@GLSxtr@p@acrlong@	105
\@GLSxtr@p@acrlongpl@	105
\@GLSxtr@p@acrshort@	105
\@GLSxtr@p@acrshortpl@	105
\@GLSxtr@p@long@	104
\@GLSxtr@p@longpl@	105
\@GLSxtr@p@plural@	104
\@GLSxtr@p@short@	104
\@GLSxtr@p@shortpl@	104
\@GLSxtr@p@text@	104
\@GLSxtrlong	104, 236
\@GLSxtrlongpl	105, 240
\@GLSxtrp	112
\@GLSxtrshort	104, 234
\@GLSxtrshortpl	104, 238
\@Gls@	104, 119, 120, 174, 388
\@Gls@acrentryname	128
\@Gls@entry@field	41, 95, 111, 208
\@Gls@entryname	128

\@GlsXtrEnableOnTheFly	66	\@cGlspl@	117, 126
\@GlsXtrIfFieldCmpNum	39, 40	\@cgls@	117, 126
\@GlsXtrIfFieldEqNum	40	\@cglspl@	117, 126
\@GlsXtrIfFieldEqStr	43	\@currentlabelname	140
\@GlsXtrIfFieldEqXpStr	44	\@dGLS	388
\@GlsXtrIfFieldNonZero	39	\@dGLSpl	388
\@GlsXtrIfFieldValueInCsvList	38	\@dGls	388
\@GlsXtrIfXpFieldEqXpStr	44	\@dGlspl	388
\@GlsXtrStartUnsetBuffering	113	\@dblfloat	19
\@GlsXtrStopUnsetBuffering	114	\@dgls	387
\@Glspl@	104, 120, 121, 174, 388	\@dglspl	388
\@Glsplural@	105	\@disable@onlypremakeg	134
\@Glsstext@	105	\@do@auxoutstuff	148, 149
\@Glsxtr	67, 68	\@do@glsgl@getcounterprefix	11, 12
\@Glsxtr@full	230	\@do@glsglsee	61, 62
\@Glsxtr@fullpl	232	\@do@newglossaryentry	129, 226, 227
\@Glsxtr@p@acrlong@	105	\@do@seeglossary	15–17, 29, 64, 134
\@Glsxtr@p@acrlongpl@	105	\@do@wrglossary	80, 172
\@Glsxtr@p@acrshort@	105	\@domakeglossaries	64, 133
\@Glsxtr@p@acrshortpl@	105	\@dtl@formatlist@handler	37
\@Glsxtr@p@long@	104	\@dtl@formatlist@itemsep	37
\@Glsxtr@p@longpl@	104	\@dtl@formatlist@lastitem	37
\@Glsxtr@p@plural@	104	\@dtl@formatlist@prelastitem	37
\@Glsxtr@p@short@	104	\@dtl@formatlist@prelastitemsep	37
\@Glsxtr@p@shortpl@	104	\@dtlformatlist	38
\@Glsxtr@p@text@	104	\@empty	32,
\@Glsxtrlong	104, 235		82, 90–94, 130, 131, 139, 147, 211, 230–240
\@Glsxtrlongpl	104, 239	\@end@glsgl@addunused	62, 63
\@Glsxtrp	111	\@end@glsgl@gettype	137, 141
\@Glsxtrpl	68	\@end@glsgl@usesee	55
\@Glsxtrshort	104, 233	\@end@glsgl@trifhyphenstart	334
\@Glsxtrshortpl	104, 237	\@endfortrue	38, 207, 243, 387
\@acrlong	105	\@firstofone	82, 160,
\@acrlongpl	105		163, 187, 201, 202, 209, 215, 216, 381, 382
\@acrshort	105	\@firstofthree	75, 82,
\@acrshortpl	105		90–93, 100, 101, 230, 231, 233, 235, 237, 239
\@addtoacronymlists	128, 130, 131	\@firstoftwo	
\@addtoreset	176		.. 83–87, 91–94, 97, 100, 132, 167, 207,
\@afterheading	425,		208, 221, 222, 230–232, 237–240, 353, 354
	426, 436, 437, 439, 441, 454–458, 466, 492	\@float	19
\@alt@glsgl@hyp@opt	100	\@for	6, 25, 38,
\@auxout	11–13, 64, 72, 73, 101, 118, 127, 134,		63, 82, 115, 116, 128, 130–132, 134, 137,
	135, 148, 149, 153, 155–157, 168, 387, 466		145, 160, 163, 171, 177, 200, 207, 215, 387
\@bibglsgl@restreat	153	\@glo@alias	60, 61
\@cGLS	121	\@glo@assign@sortkey	136
\@cGLS@	117, 121, 126	\@glo@autosee	30
\@cGLSpl	121	\@glo@autoseehook	61
\@cGLSpl@	118, 121, 126	\@glo@category	122
\@cGls@	117, 126	\@glo@check@sortallowed	137

<code>\@glo@counterprefix</code>	. 11, 12, 31, 32, 147, 382	<code>\@gls@declareoption</code> 5
<code>\@glo@countunit</code> 122	<code>\@gls@default@longpl</code> 225, 226
<code>\@glo@default@sorttype</code> 137	<code>\@gls@deffile</code> 63
<code>\@glo@desc</code> 49	<code>\@gls@doautomake</code> 137, 156
<code>\@glo@descplural</code> 49, 50	<code>\@gls@doautomake@err</code> 156
<code>\@glo@group</code> 14	<code>\@gls@enablesavenonumberlist</code> 63
<code>\@glo@label</code>	14, 34, 55, 60–62, 95, 103, 444–451	<code>\@gls@encapchar</code> 211
<code>\@glo@location</code> 14	<code>\@gls@entry@count</code> 118
<code>\@glo@loclist</code> 14	<code>\@gls@entry@field</code>
<code>\@glo@name</code> 210	. 34, 41, 42, 60, 95, 109–112, 117, 158, 159	
<code>\@glo@no@assign@sortkey</code> 141	<code>\@gls@entry@unitcount</code> 126, 127
<code>\@glo@parent</code> 446, 447	<code>\@gls@field@font</code> 82–89
<code>\@glo@see</code> 55, 58, 61, 62	<code>\@gls@field@link</code> 82–89, 96, 97
<code>\@glo@seealso</code> 60, 61	<code>\@gls@firstaccess</code> 185, 186, 190
<code>\@glo@sort</code> 210	<code>\@gls@firstpluralaccess</code> 185, 186, 190
<code>\@glo@sorttype</code> 137, 145	<code>\@gls@get@counterprefix</code> 31, 32
<code>\@glo@text</code> 75	<code>\@gls@getcounterprefix</code> 11
<code>\@glo@thislettergrp</code> 166	<code>\@gls@getgrouptitle</code> 144, 160, 163
<code>\@glo@thisvalue</code> 324	<code>\@gls@grptitle</code> 102, 145
<code>\@glo@tmp</code>	. 31, 32, 34, 58, 95, 147, 176, 382, 387	<code>\@gls@hyp@opt</code> 96, 97,
<code>\@glo@type</code>	62, 102, 129, 130, 134, 137, 138,	101, 121, 169, 173–175, 229–240, 387, 388	
141, 144, 145, 148, 149, 151, 152, 160–163		<code>\@gls@hyp@opt@cs</code> 100, 101
<code>\@glo@types</code> 15, 133, 198, 199, 444–451	<code>\@gls@ifaccessattribute@set</code> 188
<code>\@glossary@default@style</code>	<code>\@gls@ifinlist</code> 165
..... 69, 70, 138, 162, 460		<code>\@gls@increment@currcount</code> 117
<code>\@glossarystyle</code> 138, 162	<code>\@gls@increment@currunitcount</code> 125
<code>\@glossentrysymbol</code> 214	<code>\@gls@initaccesskeys</code> 224
<code>\@gls@</code> 104, 119, 120, 173, 388	<code>\@gls@keymap</code> 14, 34, 55, 60, 95, 155, 207
<code>\@gls@@automake@immediate</code> 134	<code>\@gls@label</code>	8, 9, 11, 12, 64, 100, 135, 157, 243
<code>\@gls@@link</code> 76	<code>\@gls@levelchar</code> 211
<code>\@gls@ReturnAfterFi</code> 147, 382	<code>\@gls@link</code> 35, 74–76, 90–94, 230–240
<code>\@gls@actualchar</code> 211	<code>\@gls@link@checkfirsthyper</code> 75, 132
<code>\@gls@actuaallong</code> 187, 188	<code>\@gls@link@label</code> 78, 172
<code>\@gls@actuaallongpl</code> 187, 189	<code>\@gls@link@nocheckfirsthyper</code>
<code>\@gls@actualshort</code> 187–189 74, 90–94, 230–240	
<code>\@gls@actualshortpl</code> 187–189	<code>\@gls@link@opts</code> 78
<code>\@gls@adjustmode</code> 81	<code>\@gls@list</code> 144, 145
<code>\@gls@alt@hyp@opt</code> 101	<code>\@gls@local@increment@currcount</code>	... 117
<code>\@gls@alt@hyp@opt@char</code> 100, 101	<code>\@gls@local@increment@currunitcount</code>	125
<code>\@gls@alt@hyp@opt@keys</code> 101	<code>\@gls@location</code> 167, 168
<code>\@gls@assign@actual</code> 188	<code>\@gls@loclist</code> 142, 143, 167
<code>\@gls@automake</code> 137	<code>\@gls@long</code> 225
<code>\@gls@between</code> 144, 145	<code>\@gls@longaccess</code> 186, 188
<code>\@gls@checkedmkidx</code> 211, 213, 214	<code>\@gls@longaccesspl</code> 186, 188, 189
<code>\@gls@checkmkidxchars</code> 59, 210	<code>\@gls@longpl</code> 187, 223, 225, 226
<code>\@gls@codepage</code> 149	<code>\@gls@map</code> 207
<code>\@gls@counter</code>	<code>\@gls@nameaccess</code> 185, 186, 189
..... 9, 11, 12, 27, 32, 77, 78, 81, 99, 172		<code>\@gls@nohyperlist</code> 50, 52
<code>\@gls@currentlettergroup</code>	145, 160, 163, 166	<code>\@gls@noidx@do</code> 145

<code>\@gls@noidx@getgrouptitle</code>	160, 163	<code>\@gls@longextra@begintab</code>	472–486, 488–490
<code>\@gls@noidx@nosanitizesort</code>	136	<code>\@gls@longextrawidestname</code>	470, 471
<code>\@gls@noidx@sanitizesort</code>	136	<code>\@gls@nextpages</code>	138, 162
<code>\@gls@noidx@loclist@finalsep</code>	142	<code>\@gls@nonextpages</code>	138, 162
<code>\@gls@noidx@loclist@prev</code>	142	<code>\@gls@numberformat</code>	
<code>\@gls@noidx@loclist@sep</code>	142	...	9, 11, 12, 78, 81, 99, 172, 207, 210, 382
<code>\@gls@noref@warn</code>	135, 145	<code>\@gls@order</code>	134
<code>\@gls@org@glsnoidx@displayloc</code> ..	142, 143	<code>\@gls@pl@</code>	104, 119, 120, 173, 388
<code>\@gls@org@glssseeformat</code>	142, 143	<code>\@gls@plural@</code>	105
<code>\@gls@org@longpl</code>	226	<code>\@gls@punc@token</code>	221
<code>\@gls@org@shortpl</code>	226	<code>\@gls@recordlocref</code>	11
<code>\@gls@pluralaccess</code>	185, 186, 189	<code>\@gls@showtarget</code>	103
<code>\@gls@preglossaryhook</code>	138, 163, 216	<code>\@gls@style@altlist</code>	425
<code>\@gls@prevlevel</code>	452–454, 458–460	<code>\@gls@style@altlistgroup</code>	425
<code>\@gls@quotechar</code>	211	<code>\@gls@style@altlisthypergroup</code>	426
<code>\@gls@reference</code>	64, 65, 134, 135	<code>\@gls@style@alttree</code>	441
<code>\@gls@restoreat</code>	63	<code>\@gls@style@alttreegroup</code>	453
<code>\@gls@saveentrycounter</code>	15–17, 31, 79, 81, 172	<code>\@gls@style@alttreehypergroup</code>	454
<code>\@gls@see@noindex</code>	30, 153, 154	<code>\@gls@style@index</code>	435
<code>\@gls@setdefault@glslink@opts</code>		<code>\@gls@style@indexgroup</code>	436
.....	9, 36, 79, 99	<code>\@gls@style@indexhypergroup</code>	436
<code>\@gls@setsort</code>	79, 81	<code>\@gls@style@inline</code>	434
<code>\@gls@setup@default@access</code>	226	<code>\@gls@style@list</code>	424
<code>\@gls@setupsort@none</code>	16	<code>\@gls@style@listdotted</code>	423
<code>\@gls@short</code>	225, 226	<code>\@gls@style@listgroup</code>	425
<code>\@gls@shortaccess</code>	185, 186, 188–190	<code>\@gls@style@listhypergroup</code>	425
<code>\@gls@shortaccesspl</code>	185, 186, 188–190	<code>\@gls@style@mcolalttree</code>	458
<code>\@gls@shortpl</code>	187, 222, 225, 226	<code>\@gls@style@mcolalttreegroup</code>	458
<code>\@gls@sort</code>	166	<code>\@gls@style@mcolalttreehypergroup</code> ..	459
<code>\@gls@textaccess</code>	185, 186, 189	<code>\@gls@style@mcolalttreespannav</code>	459
<code>\@gls@thisHloc</code>	31	<code>\@gls@style@mcolindexgroup</code>	454
<code>\@gls@thislabel</code>	82, 115, 387–389	<code>\@gls@style@mcolindexhypergroup</code>	455
<code>\@gls@thisloc</code>	31	<code>\@gls@style@mcolindexspannav</code>	455
<code>\@gls@thisval</code>	207	<code>\@gls@style@mcoltreegroup</code>	455
<code>\@gls@tmp</code>	44, 45, 145, 187	<code>\@gls@style@mcoltreehypergroup</code>	456
<code>\@gls@tmpb</code>	213, 214	<code>\@gls@style@mcoltreenamegroup</code>	457
<code>\@gls@type</code> ..	131, 132, 135, 137, 243, 444–451	<code>\@gls@style@mcoltreenamehypergroup</code>	457
<code>\@gls@write@entrycounts</code>	118	<code>\@gls@style@mcoltreenamepannav</code> ..	457
<code>\@gls@write@entryunitcounts</code>	126	<code>\@gls@style@mcoltreespannav</code>	456
<code>\@gls@write@entryunitcounts@do</code>	127	<code>\@gls@style@tree</code>	437
<code>\@gls@writedef</code>	64	<code>\@gls@style@treegroup</code>	439
<code>\@gls@xref</code>	13, 59	<code>\@gls@style@treehypergroup</code>	439
<code>\@gls@abbrv@current@abbreviation</code>	225, 240	<code>\@gls@style@treename</code>	439
<code>\@gls@acronymlists</code>	128, 130–132	<code>\@gls@style@treenamegroup</code>	441
<code>\@gls@doifexistsorwarn</code> ...	18, 203–206, 208	<code>\@gls@style@treenamehypergroup</code> ...	441
<code>\@gls@entry</code>	64, 118, 127	<code>\@gls@target</code>	104, 140, 141
<code>\@gls@link</code>	80, 102–104	<code>\@gls@text@</code>	105
<code>\@gls@localreset</code>	115	<code>\@gls@unset</code>	113, 114
<code>\@gls@localunset</code>	114, 115	<code>\@gls@widestname</code>	444, 452

<code>\@glsxtr</code>	67, 68	<code>\@glsxtr@csname</code>	123–126
<code>\@glsxtr@do@wrglossary</code>	135	<code>\@glsxtr@current@style</code>	69, 460
<code>\@glsxtr@abbreviationsdef</code>	21, 30, 31	<code>\@glsxtr@currentunitcount</code> ..	123, 125, 126
<code>\@glsxtr@abbrlists</code>	130	<code>\@glsxtr@currunitcount</code>	124, 127
<code>\@glsxtr@accessdisplay</code>	207–209	<code>\@glsxtr@debugnr</code>	28
<code>\@glsxtr@acronymlists</code>	131	<code>\@glsxtr@debugval</code>	28
<code>\@glsxtr@activate@initialtagging</code> ..		<code>\@glsxtr@declareoption</code> 5, 19–21, 24, 25, 27	
.....	215, 217	<code>\@glsxtr@defaultnoglossarywarning</code> ..	25
<code>\@glsxtr@addabbreviationlist</code>	227	<code>\@glsxtr@defaultnumberformat</code>	
<code>\@glsxtr@addunitcounter</code>	122	7, 9, 78, 81, 99, 207, 210
<code>\@glsxtr@addunused</code>	63	<code>\@glsxtr@defpostpunc</code>	20, 28
<code>\@glsxtr@addunusedxrefs</code>	62, 63	<code>\@glsxtr@deprecated@abbrstyle</code>	
<code>\@glsxtr@altmodifier</code>	101	275, 277, 279,
<code>\@glsxtr@assign@leveloffset</code>	140	281, 292, 294, 295, 298, 313, 316, 320, 323	
<code>\@glsxtr@attrval</code> . 80, 201–206, 208, 210, 214		<code>\@glsxtr@dialect</code>	45, 46
<code>\@glsxtr@autoindex@at</code>	210–212	<code>\@glsxtr@disabledflycommand</code>	68
<code>\@glsxtr@autoindex@doextra@esc</code>	210	<code>\@glsxtr@display@loc</code>	146
<code>\@glsxtr@autoindex@encap</code>	210–212	<code>\@glsxtr@do@@wrintex</code>	100
<code>\@glsxtr@autoindex@esc</code>	211, 213, 214	<code>\@glsxtr@do@autoadd</code>	78
<code>\@glsxtr@autoindex@escat</code>	211, 212	<code>\@glsxtr@do@glsglisablehyperinlist</code> ..	98
<code>\@glsxtr@autoindex@escencap</code> ...	211, 212	<code>\@glsxtr@do@inc@linkcount</code>	177
<code>\@glsxtr@autoindex@esclevel</code> ...	211, 212	<code>\@glsxtr@do@nameref@record</code>	11, 12
<code>\@glsxtr@autoindex@escquote</code> ...	211, 213	<code>\@glsxtr@do@record@wrglossary</code>	8, 15
<code>\@glsxtr@autoindex@level</code>	211, 212	<code>\@glsxtr@do@redef@forglsentries</code>	7
<code>\@glsxtr@autoindex@setname</code>	210	<code>\@glsxtr@do@style</code>	26, 376
<code>\@glsxtr@autoindex@crossrefs</code> 15, 18, 55, 60		<code>\@glsxtr@do@titlecaps@warn</code>	
<code>\@glsxtr@autoseeindexfalse</code>	15	202, 203, 205, 208, 216
<code>\@glsxtr@autoseeindextrue</code>	19	<code>\@glsxtr@doabbreviationsdef</code>	21
<code>\@glsxtr@base@acrcmd</code>	90–95, 130–132	<code>\@glsxtr@doaccsupp</code>	24, 28
<code>\@glsxtr@bibgls@removespaces</code>	382	<code>\@glsxtr@docdefsetting</code>	18, 64
<code>\@glsxtr@bookindex@atendgroup</code>		<code>\@glsxtr@docdefval</code>	18, 63–65
.....	463, 464, 466	<code>\@glsxtr@docounterrecord</code>	13
<code>\@glsxtr@bookindex@atsubendgroup</code> ..	465	<code>\@glsxtr@doglossary</code>	160, 161, 163, 164
<code>\@glsxtr@bookindex@atsubsubendgroup</code>	465	<code>\@glsxtr@doiflabelinlist</code>	165
<code>\@glsxtr@bookindex@between</code> . 463, 464, 466		<code>\@glsxtr@doloadprefix</code>	25, 28, 31
<code>\@glsxtr@bookindex@sep</code>	463–465	<code>\@glsxtr@doloctag</code>	71, 72
<code>\@glsxtr@bookindex@subatendgroup</code> ..		<code>\@glsxtr@dorecord</code>	8, 10
.....	463, 464, 466	<code>\@glsxtr@dorecordnodefer</code>	8, 10
<code>\@glsxtr@bookindex@subbetween</code> . 463–465		<code>\@glsxtr@dosee@alsoindex@glossary</code> ..	17
<code>\@glsxtr@bookindex@subsep</code>	463–465	<code>\@glsxtr@doseeglossary</code>	16, 30
<code>\@glsxtr@bookindex@subsubatendgroup</code>		<code>\@glsxtr@dostylewarn</code>	243
.....	464, 466	<code>\@glsxtr@enabletagging</code>	215
<code>\@glsxtr@bookindex@subsubbetween</code> ..		<code>\@glsxtr@end@</code>	66
.....	463–465	<code>\@glsxtr@endescpsh</code>	211–214
<code>\@glsxtr@bookindex@groupskip</code> ...	464, 466	<code>\@glsxtr@entrycount@org@localreset</code>	117
<code>\@glsxtr@cat</code>	116, 128, 171, 215	<code>\@glsxtr@entrycount@org@localunset</code>	117
<code>\@glsxtr@check@bibgls@nameref</code>	153	<code>\@glsxtr@entrycount@org@reset</code>	117
<code>\@glsxtr@checkgroup</code>	161, 164	<code>\@glsxtr@entrycount@org@unset</code>	117
<code>\@glsxtr@counterrecordhook</code>	11–13		

<code>\@glsxtr@entryunitcount@org@localreset</code>	<code>\@glsxtr@nomissingglstextnr</code>	25
..... 125	<code>\@glsxtr@nomissingglstextval</code>	25
<code>\@glsxtr@entryunitcount@org@localunset</code>	<code>\@glsxtr@noop@recordcounter</code>	13, 16
..... 125	<code>\@glsxtr@nopostpunc</code>	139
<code>\@glsxtr@entryunitcount@org@reset</code>	<code>\@glsxtr@nopostpunc@postdesc</code>	139
..... 125	<code>\@glsxtr@notfoundinlist</code>	221
<code>\@glsxtr@equationsfalse</code>	<code>\@glsxtr@op@recordcounter</code>	15, 17
..... 19	<code>\@glsxtr@optlist</code>	68
<code>\@glsxtr@err@undefaction</code>	<code>\@glsxtr@org@starttoc</code>	353
..... 7, 16	<code>\@glsxtr@org@GLS@</code>	75
<code>\@glsxtr@field@linkdefs</code>	<code>\@glsxtr@org@GLSpl@</code>	75
..... 74, 76	<code>\@glsxtr@org@Gls@</code>	75
<code>\@glsxtr@floatsfalse</code>	<code>\@glsxtr@org@Glspl@</code>	75
..... 19	<code>\@glsxtr@org@Glsxtrtitlefirst</code>	354, 355
<code>\@glsxtr@format@overridefalse</code>	<code>\@glsxtr@org@Glsxtrtitlefirstplural</code>	354, 355
..... 209 354, 355	
<code>\@glsxtr@format@overridetrue</code>	<code>\@glsxtr@org@Glsxtrtitlefull</code>	354, 356
..... 209	<code>\@glsxtr@org@Glsxtrtitlefullpl</code>	354, 356
<code>\@glsxtr@foundinlist</code>	<code>\@glsxtr@org@Glsxtrtitlelong</code>	354, 355
..... 222	<code>\@glsxtr@org@Glsxtrtitlelongpl</code>	354, 355
<code>\@glsxtr@full</code>	<code>\@glsxtr@org@Glsxtrtitlename</code>	354, 355
..... 229	<code>\@glsxtr@org@Glsxtrtitleplural</code>	354, 355
<code>\@glsxtr@fullpl</code>	<code>\@glsxtr@org@Glsxtrtitleshort</code>	354, 355
..... 231	<code>\@glsxtr@org@Glsxtrtitleshortpl</code>	354, 355
<code>\@glsxtr@get@prefixedlabel</code>	<code>\@glsxtr@org@Glsxtrtitletext</code>	354, 355
.... 388, 389	<code>\@glsxtr@org@MakeUppercase</code>	354, 355
<code>\@glsxtr@gettype</code>	<code>\@glsxtr@org@addtoacronymlists</code>	128, 131
..... 137	<code>\@glsxtr@org@checkfirsthyper</code>	97, 132
<code>\@glsxtr@glossdescfont</code>	<code>\@glsxtr@org@currentfieldvalue</code>	45
..... 201–203	<code>\@glsxtr@org@delimN</code>	72
<code>\@glsxtr@glossnamefont</code>	<code>\@glsxtr@org@delimR</code>	72
..... 203–206, 208, 209	<code>\@glsxtr@org@doseeglossary</code>	29, 135
<code>\@glsxtr@glosssymbolfont</code>	<code>\@glsxtr@org@gloautosee</code>	30
..... 214, 215	<code>\@glsxtr@org@glolinkprefix</code>	78, 80
<code>\@glsxtr@gobbleto@endescspch</code>	<code>\@glsxtr@org@gls@</code>	74
..... 213	<code>\@glsxtr@org@glsdohypertarget</code>	141
<code>\@glsxtr@groupheading</code>	<code>\@glsxtr@org@glsignore</code>	72
..... 161, 164, 166	<code>\@glsxtr@org@glspl@</code>	74, 75
<code>\@glsxtr@idx@displaynumberlist</code>	<code>\@glsxtr@org@glsxtrtitlefirst</code>	354, 355
.... 135	<code>\@glsxtr@org@glsxtrtitlefirstplural</code>	354, 355
<code>\@glsxtr@idx@entrynumberlist</code> 354, 355	
..... 136	<code>\@glsxtr@org@glsxtrtitlefull</code>	354, 355
<code>\@glsxtr@if@record@only</code>	<code>\@glsxtr@org@glsxtrtitlefullpl</code>	354, 356
..... 133, 156	<code>\@glsxtr@org@glsxtrtitlelong</code>	354, 355
<code>\@glsxtr@ifcsstart</code>	<code>\@glsxtr@org@glsxtrtitlelongpl</code>	354, 355
..... 66	<code>\@glsxtr@org@glsxtrtitlename</code>	354, 355
<code>\@glsxtr@ifischild</code>	<code>\@glsxtr@org@glsxtrtitleplural</code>	354, 355
..... 167	<code>\@glsxtr@org@glsxtrtitleshort</code>	354, 355
<code>\@glsxtr@ifnum@mmode</code>	<code>\@glsxtr@org@glsxtrtitleshortpl</code>	354, 355
..... 77	<code>\@glsxtr@org@glsxtrtitletext</code>	354, 355
<code>\@glsxtr@ifpunctoken</code>	<code>\@glsxtr@org@MakeUppercase</code>	354, 355
..... 221	<code>\@glsxtr@org@addtoacronymlists</code>	128, 131
<code>\@glsxtr@ifunitcounter</code>	<code>\@glsxtr@org@checkfirsthyper</code>	97, 132
..... 122	<code>\@glsxtr@org@currentfieldvalue</code>	45
<code>\@glsxtr@insert@dots</code>	<code>\@glsxtr@org@delimN</code>	72
..... 223	<code>\@glsxtr@org@delimR</code>	72
<code>\@glsxtr@insert@dots@next</code>	<code>\@glsxtr@org@doseeglossary</code>	29, 135
..... 223	<code>\@glsxtr@org@gloautosee</code>	30
<code>\@glsxtr@insertdots</code>	<code>\@glsxtr@org@glolinkprefix</code>	78, 80
..... 188, 225	<code>\@glsxtr@org@gls@</code>	74
<code>\@glsxtr@label</code>	<code>\@glsxtr@org@glsdohypertarget</code>	141
..... 38, 63, 177, 200, 201	<code>\@glsxtr@org@glsignore</code>	72
<code>\@glsxtr@labelprefixes</code>	<code>\@glsxtr@org@glspl@</code>	74, 75
..... 386, 387	<code>\@glsxtr@org@glsxtrtitlefirst</code>	354, 355
<code>\@glsxtr@leveloffset</code>	<code>\@glsxtr@org@glsxtrtitlefirstplural</code>	354, 355
..... 141, 167 354, 355	
<code>\@glsxtr@loadstyles</code>	<code>\@glsxtr@org@glsxtrtitlefull</code>	354, 355
..... 422, 423	<code>\@glsxtr@org@glsxtrtitlefullpl</code>	354, 356
<code>\@glsxtr@local@textformat</code>	<code>\@glsxtr@org@glsxtrtitlelong</code>	354, 355
..... 78–80	<code>\@glsxtr@org@glsxtrtitlelongpl</code>	354, 355
<code>\@glsxtr@locale</code>	<code>\@glsxtr@org@glsxtrtitlename</code>	354, 355
..... 45, 46	<code>\@glsxtr@org@glsxtrtitleplural</code>	354, 355
<code>\@glsxtr@longnewglossaryentry</code>	<code>\@glsxtr@org@glsxtrtitleorpdforheading</code>	354, 355
..... 49 354, 355	
<code>\@glsxtr@mark@wordseps</code>	<code>\@glsxtr@org@glsxtrtitleplural</code>	354, 355
..... 224 354, 355	
<code>\@glsxtr@mark@wordseps@next</code> 354, 355	
..... 224		
<code>\@glsxtr@markwordseps</code>		
..... 225, 226		
<code>\@glsxtr@mixed@assign@sortkey</code>		
..... 136		
<code>\@glsxtr@newglslslike</code>		
..... 168		
<code>\@glsxtr@noidx@displaynumberlist</code>		
.. 136		
<code>\@glsxtr@noidx@do</code>		
..... 166		
<code>\@glsxtr@noidx@entrynumberlist</code>		
.... 136		
<code>\@glsxtr@noidx@numberlistloop</code>		
..... 136		

<code>\@glxtr@org@glxtrtitleshort</code>	354, 355	<code>\@glxtr@record@setting@off</code>	64, 101, 168
<code>\@glxtr@org@glxtrtitleshortpl</code>	354, 355	<code>\@glxtr@record@setting@only</code>	15, 32
<code>\@glxtr@org@glxtrtitletext</code>	354, 355	<code>\@glxtr@recordsee</code>	15, 29, 59
<code>\@glxtr@org@makeglossaries</code>	132, 133	<code>\@glxtr@redef@for@gl@sentries</code>	7, 30, 31
<code>\@glxtr@org@markboth</code>	353	<code>\@glxtr@redefstyles</code>	25, 26, 376
<code>\@glxtr@org@markright</code>	352, 353	<code>\@glxtr@reg@glosslist</code>	134–137, 141
<code>\@glxtr@org@newacronymstyle</code>	130–132	<code>\@glxtr@restore@postpunc</code>	139
<code>\@glxtr@org@postdescription</code>	139, 217	<code>\@glxtr@rglstrigger@record</code>	173–175
<code>\@glxtr@org@see@noindex</code>	153, 154	<code>\@glxtr@s@longnewglossaryentry</code>	49
<code>\@glxtr@org@setacronymlists</code>	131	<code>\@glxtr@save@preloctag</code>	71–73
<code>\@glxtr@org@setacronymstyle</code>	130, 132	<code>\@glxtr@setentrycount@unsetattr</code>	116
<code>\@glxtr@org@theHvalue</code>	8–10	<code>\@glxtr@setentryunitcount@unsetattr</code>	128
<code>\@glxtr@org@unset@buffer</code>	113, 114	<code>\@glxtr@setupshortcuts</code>	23, 24, 30, 31
<code>\@glxtr@org@prefix</code>	11	<code>\@glxtr@shortcutsnr</code>	23
<code>\@glxtr@org@printglossary</code>	68, 139	<code>\@glxtr@shortcutsval</code>	23, 156
<code>\@glxtr@org@warndep</code>	225	<code>\@glxtr@swaptwo</code>	222
<code>\@glxtr@p@acrlong@</code>	105	<code>\@glxtr@tag</code>	215
<code>\@glxtr@p@acrlongpl@</code>	105	<code>\@glxtr@taggingcs</code>	215
<code>\@glxtr@p@acrshort@</code>	105	<code>\@glxtr@textformat</code>	80
<code>\@glxtr@p@acrshortpl@</code>	105	<code>\@glxtr@theHentrycounter</code>	11
<code>\@glxtr@p@long@</code>	104	<code>\@glxtr@theHvalue</code>	8–10, 77, 79, 81, 172
<code>\@glxtr@p@longpl@</code>	104	<code>\@glxtr@theentrycounter</code>	11
<code>\@glxtr@p@plural@</code>	104	<code>\@glxtr@thevalue</code>	8–10, 77, 79, 81, 172
<code>\@glxtr@p@short@</code>	104	<code>\@glxtr@thisloctag</code>	72
<code>\@glxtr@p@shortpl@</code>	104	<code>\@glxtr@titlelabel</code>	143, 144, 166
<code>\@glxtr@p@text@</code>	104	<code>\@glxtr@tmp</code>	25, 26, 77, 146
<code>\@glxtr@pagetag</code>	71, 72	<code>\@glxtr@type</code>	200
<code>\@glxtr@pagetag</code>	71, 72	<code>\@glxtr@unitcountlist</code>	122, 123
<code>\@glxtr@prefix</code>	387	<code>\@glxtr@unset</code>	113, 114
<code>\@glxtr@prevunitcount</code>	125	<code>\@glxtr@unset@buffer</code>	113, 114
<code>\@glxtr@printglossnr</code>	140	<code>\@glxtr@unsrt@getgroup@title</code>	160, 163
<code>\@glxtr@printglossopts</code>	68, 137, 139, 161, 162	<code>\@glxtr@use@equation@counter</code>	9, 77, 79
<code>\@glxtr@printglossval</code>	140	<code>\@glxtr@use@see</code>	55
<code>\@glxtr@printunsrtglossaryskipentry</code>	161, 163, 164	<code>\@glxtr@warn@hybrid@noprintgloss</code>	133
<code>\@glxtr@provide@addstoragekey</code>	35	<code>\@glxtr@warn@onexists@ordo</code>	7, 15, 17
<code>\@glxtr@provide@storagekey</code>	34	<code>\@glxtr@warn@undefaction</code>	7, 15, 17
<code>\@glxtr@providenewgls</code>	168	<code>\@glxtr@wrglossary@location@hyperlink</code>	27
<code>\@glxtr@record</code>	15–17, 74–76, 81, 230–240	<code>\@glxtr@wrglossnr</code>	76
<code>\@glxtr@record@noglossarywarning</code>	16	<code>\@glxtr@wrglossval</code>	76
<code>\@glxtr@record@only@setup</code>	16, 17	<code>\@glxtr@buffer@nodup@unset</code>	114
<code>\@glxtr@record@setting</code>	8, 10–12, 15, 16, 32, 59, 64, 65, 101, 133, 134, 151, 153–156, 168	<code>\@glxtr@buffer@unset</code>	113
<code>\@glxtr@record@setting@alsoindex</code>	8, 10, 16, 17, 59, 133, 134	<code>\@glxtr@dialecthook</code>	376
<code>\@glxtr@record@setting@nameref</code>	11, 12, 15, 32, 154, 155	<code>\@glxtr@docdef@false</code>	65
		<code>\@glxtr@entryfmt</code>	36
		<code>\@glxtr@fmt</code>	35
		<code>\@glxtr@glossentry</code>	157
		<code>\@glxtr@glossentry@other</code>	158, 159
		<code>\@glxtr@hypername@prefix</code>	140, 141, 165

<code>\@glxtrifhasfield</code>	38, 39, 43, 44	<code>\@no@glxtrindexaliased</code>	98, 99
<code>\@glxtrifhyphenstart</code>	334	<code>\@no@makeglossaries</code>	152
<code>\@glxtrindexaliased</code>	98	<code>\@nocounterr</code>	177
<code>\@glxtrindexcrossrefsfalse</code>	18	<code>\@nopostdesc</code>	139, 162
<code>\@glxtrindexcrossrefstrue</code>	18	<code>\@onelevel@sanitize</code> .	11, 13, 59, 68, 144, 166
<code>\@glxtrinmark</code>	353	<code>\@onlypreamble</code>	69,
<code>\@glxtrlong</code>	104, 235		72, 126, 154, 157, 177, 209, 212, 213, 215
<code>\@glxtrlongpl</code>	104, 238	<code>\@org@glossaryentrynumbers</code> .	138, 162, 163
<code>\@glxtrnewgls</code>	169, 170	<code>\@org@newglossaryentryprehook</code>	49
<code>\@glxtrnewgls@inner</code>	168, 169	<code>\@print@unsrt@glossary</code>	160
<code>\@glxtrnewgls@innercsname</code>	169	<code>\@print@unsrt@innerglossary</code>	161
<code>\@glxtrnotinmark</code>	353	<code>\@printgloss@checkexists</code>	137, 159
<code>\@glxtrp</code>	110	<code>\@printgloss@checkexists@allowignored</code>	
<code>\@glxtrp@opt</code>	108		159
<code>\@glxtrpl</code>	67, 68	<code>\@printgloss@setsort</code>	136, 138, 162
<code>\@glxtrpostloctag</code>	71, 73	<code>\@printglossary</code>	68, 160
<code>\@glxtrpreloctag</code>	71, 73	<code>\@printunsrt@glossary@handler</code> .	161, 164
<code>\@glxtrsetaliasnoindex</code>	98, 99	<code>\@printunsrtglossary</code>	159
<code>\@glxtrshort</code>	104, 233	<code>\@rGLS</code>	174
<code>\@glxtrshortpl</code>	104, 237	<code>\@rGLS@</code>	174
<code>\@glxtrundeftag</code>	6, 32	<code>\@rGLSpl</code>	175
<code>\@glxtrwrglossmark</code>	27, 28	<code>\@rGLSpl@</code>	175
<code>\@gobble</code>	7,	<code>\@rGls</code>	173
	16, 19, 82, 131, 134, 164, 165, 223, 463–466	<code>\@rGls@</code>	173
<code>\@gobbletwo</code>	132, 225	<code>\@rGlspl</code>	174
<code>\@gtempa</code>	378	<code>\@rGlspl@</code>	174
<code>\@ifdefinable</code>	378	<code>\@rc@ifdefinable</code>	378
<code>\@ifglossaryexists</code>	33	<code>\@rgls</code>	173
<code>\@ifnextchar</code>	100, 140, 141	<code>\@rgls@</code>	173
<code>\@ifpackageloaded</code>	5, 21, 156,	<code>\@rglspl</code>	173
	178, 201, 203, 205, 207, 209, 376, 390, 421	<code>\@rglspl@</code>	173
<code>\@ifstar</code>	33–35, 38–40,	<code>\@sGlsXtrEnableOnTheFly</code>	66
	43, 44, 49, 50, 52, 66, 100, 113, 114, 159, 215	<code>\@secondofthree</code>	83–85,
<code>\@ifundefined</code>	375, 378		90–92, 94, 96, 230, 232, 234, 236, 237, 239
<code>\@ignored@glossaries</code>	50–52	<code>\@secondoftwo</code>	
<code>\@input</code>	153		75, 82, 85–94, 97, 104, 132, 140,
<code>\@input@</code>	148		167, 207, 222, 230, 231, 233–240, 255,
<code>\@istfilename</code>	134		257, 280, 282, 297, 299, 322, 324, 354, 355
<code>\@makeglossaries@warn@noprintglossary</code>		<code>\@sglsxtr@provide@storagekey</code>	34
	135	<code>\@star@or@long</code>	378
<code>\@makeglossary</code>	134	<code>\@starttoc</code>	353
<code>\@mfu@domakefirstuc</code>	216	<code>\@thirdofthree</code>	83–85,
<code>\@mfu@nocaplist</code>	216		91–94, 97, 231, 232, 234, 236, 238, 240, 354
<code>\@ne</code>	118, 127, 169	<code>\@thirdoftwo</code>	85–89
<code>\@newglossaryentry@defcounters</code>	116, 124	<code>\@this@key</code>	207
<code>\@newglossaryentryposthook</code> .	14, 34, 60, 95	<code>\@tracklang@lang</code>	46
<code>\@newglossaryentryprehook</code>	14, 34, 49, 60, 95	<code>\@warn@nomakeglossaries</code>	149
<code>\@nil</code>	147, 166, 382	<code>\@xdy@main@language</code>	148
<code>\@nnil</code>	211, 213, 214, 221–224	<code>\@xdycrossrefhook</code>	59

`\acsp` 22
`\actualchar` 213
`\addtolength` 453
`\advance` 118, 127, 141, 154, 169
`\AF` 22
`\Af` 22
`\af` 21
`\AFP` 22
`\Afp` 22
`\afp` 21
`\AL` 22
`\Al` 22
`\al` 21
`\ALP` 22
`\Alp` 22
`\alp` 21
`\alsoname` 58, 59
`amsmath` package 12, 27
`\AnyTrackedLanguages` 376, 421
`\appto` 14, 26, 34, 55, 59–61, 95, 100, 116, 124,
161, 164, 165, 209, 221, 223–225, 386, 422
`\apptoglossarypreamble` 383–385
`\arabic` 27, 466
`\AS` 22
`\As` 21
`\as` 21
`\ASP` 22
`\Asp` 21
`\asp` 21
`\AtBeginDocument` 27, 32, 70, 156, 387
`\AtEndDocument` 62, 118, 126, 148, 149

B

`babel` package 210, 212, 221
`\begin` 17, 133, 145,
150, 151, 160, 163, 424, 426–433, 455,
456, 458, 459, 463, 472–486, 488, 489, 496
`\begingroup` 8, 9, 35–37, 78, 81, 99, 157, 159–
161, 176, 187, 214, 378, 381, 382, 387, 492
`\bgroup` 49, 138
`bib2gls` 27, 36, 48, 97, 99,
101, 153, 154, 161, 163, 166, 168, 172,
375, 376, 378, 380, 383, 385, 387, 389, 497
`\bibglshdelimN` 377
`\bibglshrefchar` 154
`\bibglslastDelimN` 377
`\bottomrule` 470, 474–
476, 478, 479, 481, 483, 484, 486, 487, 489

C

`\c@wrglossary` 27
`\catcode` 63, 153
category attributes:
`accessinsertdots` 188
`apospplural` 225
`discardperiod` 220
`entrycount` 113, 116, 118, 127, 128
`externallocation` 380
`firstshortaccess` 190, 252, 254
`firsttuc` 205
`glossdesc` 201
`glossdescfont` 201
`glossname` 203
`glossnamefont` 203, 205
`headuc` 356
`indexname` 210
`indexonlyfirst` 99
`insertdots` 225
`linkcount` 176
`linkcountmaster` 176
`markshortwords` 225
`markwords` 225, 226, 334, 335, 344
`nameshortaccess` 189, 190
`nohyper` 97
`nohyperfirst` 83–85
`noshortplural` 225, 226
`regular` 73, 121,
246–249, 251, 252, 254, 255, 257, 259,
261, 262, 264, 265, 268, 270, 271, 273,
275, 277, 279, 280, 282, 284–286, 288,
289, 292–294, 296–298, 301–308, 310,
312, 314–316, 318, 319, 321, 323, 325,
332, 334, 336–338, 340, 344, 345, 350, 352
`textformat` 79
`textshortaccess` 189
`\cdot` 27
`\centering` 462
`\cGLS` 22, 116, 127
`\cGls` 21, 22, 116, 127
`\cgls` 21, 22, 116, 127
`\cGLSformat` 120
`\cGlsformat` 119
`\cglsformat` 119, 121
`\cGLSpl` 22, 116, 128
`\cGlspl` 21, 22, 116, 127
`\cglspl` 21, 22, 116, 127
`\cGLSplformat` 120
`\cGlsplformat` 120

<code>\cglspformat</code>	119, 121	<code>\def</code>	9, 11, 13–18, 29, 31, 32, 35, 37, 40, 41, 49, 51, 55, 59, 60, 63, 66–68, 71, 73–79, 81–95, 101, 103–107, 113, 119–121, 129, 135, 137–142, 144–148, 160–163, 166, 169, 172–175, 185–188, 211–216, 221–226, 229–240, 243, 334, 335, 337, 340, 344, 346, 380, 382, 386, 387, 421, 435, 452–454, 458–460, 463, 464, 470, 471, 491, 495, 496
<code>\changes</code>	133	<code>\defglentryfmt</code>	50–53
<code>\char</code>	143	<code>\define@boolkey</code>	18, 19, 77, 98, 141
<code>\columnwidth</code>	70	<code>\define@choicekey</code>	7, 16, 18, 20, 23, 25, 28, 76, 140
<code>\count@</code>	118, 127	<code>\define@key</code>	14, 20, 25, 26, 34, 60, 77, 81, 95, 140, 185, 186, 222, 223
<code>\cs</code>	133	<code>\DefineAcronymSynonyms</code>	23
<code>\csappto</code>	47	<code>\delimN</code>	72
<code>\csdef</code>	34, 42, 47, 48, 95–97, 117, 122, 123, 126, 190, 195, 207, 218, 219, 244, 245, 255, 257, 280, 282, 297, 298, 321, 323, 326, 328, 329, 331, 341, 343, 347, 349, 423, 443	<code>\delimR</code>	72
<code>\cseappto</code>	53	<code>\descriptionname</code>	470, 473, 475, 476, 478, 479, 481, 482, 484, 486, 487, 489
<code>\csedef</code>	124, 494	<code>\detokenize</code>	66
<code>\csgdef</code>	43, 50–52, 65, 71, 117, 118, 123, 125, 126, 131, 443, 466	<code>\dimen@</code>	132, 443–451, 471
<code>\cslet</code>	42, 49, 50, 138	<code>\dimen@i</code>	446, 447
<code>\csletcs</code>	42, 43, 245	<code>\dimen@ii</code>	443, 444, 446, 447, 471
<code>\csname</code>	6, 34, 51, 59, 64, 69, 73, 75, 78, 81, 90–97, 99, 108, 123, 124, 135, 145, 148, 151, 152, 160, 162, 163, 169–172, 177, 201, 225, 230–240, 246, 452	<code>\dimexpr</code>	70, 442, 471, 472, 493
<code>\cspreto</code>	48	<code>\disable@keys</code>	21, 31, 32, 65, 153
<code>\csuse</code>	9, 12, 36, 42, 48, 51, 61, 62, 72, 96, 109–111, 122–126, 138, 143, 145, 146, 157, 158, 165–167, 195, 207, 217–219, 244, 245, 381, 383, 386, 443, 444, 446, 447, 470, 494	<code>\do</code>	6, 25, 38, 63, 82, 115, 116, 128, 130–132, 134, 137, 145, 160, 163, 171, 177, 200, 207, 215, 387
<code>\csxdef</code>	55, 60, 123, 126	<code>\do@gl@link@checkfirsthyper</code>	35, 74–76, 79, 90–94, 230–240
<code>\currentglossary</code>	138, 158, 159, 161, 162, 463, 466	<code>\do@gl@disablehyperinlist</code>	79, 98
<code>\CurrentOption</code>	28, 422, 423	<code>doc package</code>	213
<code>\CurrentTrackedLanguage</code>	421	<code>\dolistcsloop</code>	37
<code>\CurrentTrackedLanguageTag</code>	155	<code>\DTLifinlist</code>	38, 39, 78, 130, 135, 136, 141, 387
<code>\CurrentTrackedScript</code>	421	<code>\DTLifint</code>	143
<code>\CurrentTrackedTag</code>	376, 421		
<code>\CustomAbbreviationFields</code>	227, 247–250, 252, 254, 256, 258, 260, 263, 265, 266, 268–270, 272, 274, 277, 279, 280, 282–288, 290, 294, 296, 298, 300–303, 305, 307, 309, 311, 313, 316, 319–321, 323, 325, 326, 328, 329, 331–333, 335, 337–339, 341, 343–346, 348, 350, 351		
		E	
		<code>\eappto</code>	26, 188–190, 422
		<code>\edef</code>	11, 46, 63, 143, 147–149, 153, 176, 211, 213, 214, 382, 387, 421, 463–465, 472–486, 488, 489, 496
		<code>\eglssetwidest</code>	445–451
		<code>\egroup</code>	49, 50, 138
		<code>\else</code> ...	8–13, 15, 18–21, 23–25, 29–32, 36, 40, 41, 47, 63, 65, 66, 71, 73, 75, 80, 99, 100, 119, 130, 132, 134, 136, 138–140, 143, 146, 147, 150, 152, 154, 162, 167, 172, 209–211, 213, 221–224, 226, 233–240, 242, 243, 247, 248, 250, 252, 253,
D			
<code>datatool-base package</code>	12		
<code>\DeclareAcronymList</code>	129		
<code>\DeclareOption</code>	5, 422		
<code>\DeclareOptionX</code>	5, 28		

255, 256, 258–264, 267, 269, 271–279,
 281, 284, 286–295, 297, 298, 300–304,
 306, 308–320, 322, 323, 326–328, 330,
 333–335, 337, 340–344, 346, 348, 350,
 351, 382–386, 424, 427–434, 436, 439,
 440, 452, 453, 460, 462–465, 473–490, 495
 \emph 187, 299
 \empty 146, 147, 187, 382, 386, 387
 \encapchar 213
 \end 17, 133, 145, 150, 151,
 161, 163, 424, 426–433, 455, 456, 458,
 459, 464, 473–480, 482, 483, 485–490, 496
 \end@getprefix 31, 32
 \end@glstr@display@loc 146
 \endcsname 6, 34, 51, 59, 64, 69,
 73, 75, 78, 81, 90–97, 99, 108, 123, 124,
 135, 145, 148, 151, 152, 160, 162, 163,
 169–172, 177, 201, 225, 230–240, 246, 452
 \endfoot 470, 473, 475,
 476, 478, 479, 481, 482, 484, 485, 487, 489
 \endgroup 8, 10, 36, 37, 78, 82, 99,
 158–161, 176, 187, 215, 378, 382, 387, 492
 \endhead 470, 473, 475,
 476, 478, 479, 481, 482, 484, 485, 487, 489
 \ensuremath 27
 entry categories:
 abbreviation 225, 240
 general 195, 197
 index 199
 nameshortaccess 309
 \entryname 470, 471, 473, 475,
 476, 478, 479, 481, 482, 484, 486, 487, 489
 \epreto 210
 \equal 152, 219
 equation (counter) 19, 77
 \escapechar 378
 etoolbox package 5
 \everypar 493
 \expandafter 28, 34,
 35, 38, 39, 45, 55, 58, 62, 63, 66–68, 77,
 95–97, 100, 101, 108, 114, 121, 122, 130,
 135–137, 141, 145–147, 160–164, 166,
 167, 169, 176, 188, 201, 204, 205, 207,
 209, 210, 213, 221, 225, 226, 334, 378, 382
 \expandonce 129, 166,
 187–189, 211, 248, 338, 472–486, 488–490
 \ExtraCustomAbbreviationFields
 188–190, 224, 227

F

\fi 7–13, 15, 17–21, 23–25, 28–30, 32,
 36, 40, 41, 47, 55, 59, 60, 62, 64–66, 69–
 71, 73, 76, 79, 80, 99, 100, 118, 119, 126,
 127, 130, 132, 134, 136–141, 144, 146,
 147, 149, 150, 152, 154–156, 161, 162,
 164, 167, 172, 209–211, 214, 222–224,
 226, 233–240, 242, 243, 247, 248, 250,
 252, 253, 255, 256, 258–264, 267, 269,
 271–279, 281, 284–295, 297, 298, 300–
 304, 306, 308–320, 322, 323, 326–328,
 330, 333–335, 337, 340–344, 346, 348,
 350, 351, 376, 382–386, 424, 427–434,
 436, 438–440, 443–453, 460, 462–465,
 471, 473–485, 487, 488, 490, 491, 495, 496
 first use 497
 flag 497
 text 497
 \firstacronymfont 131, 132
 fontspec package 155
 \footnote 251
 \forallallabbreviationlists 130
 \forallallglossaries 62, 160, 198–200, 444–451
 \forallallglsentries 64, 118, 127
 \ForEachTrackedDialect 376, 421
 \foreignlanguage 45, 46
 \forlgsentries 6, 62, 198–200, 444–451
 \forlistcsloop 37, 127, 145
 \forlistloop 114, 142, 216
 \futurelet 221

G

\gdef 72, 212, 213
 \Genacrfullformat 129
 \genacrfullformat 129
 \GenericAcronymFields 129
 \Genplacrfullformat 129
 \genplacrfullformat 129
 \GetTrackedDialectFromLanguageTag .. 45
 \GetTrackedDialectToMapping 46
 \glo@grabfirst 166
 \glo@name 204, 205, 208, 209
 \gloaliaslabel 103
 \global 11, 49, 50, 63, 138, 163, 167
 \glolinkprefix 77, 78, 80, 103, 140, 156
 glossaries package 12, 16, 28, 30, 31, 47,
 48, 55, 59–61, 133, 134, 137, 159, 186, 423
 glossaries-accsupp package 24, 28, 178, 187, 226
 glossaries-extra package 2, 48, 421

glossaries-extra-bib2gls package	treegroup	439
..... 16, 32, 376, 421, 471	treehypergroup	439
glossaries-extra-stylemods package	treenoname	439
..... 25, 217, 376, 384	treenonamegroup	441
glossaries-prefix package	treenonamehypergroup	441
25, 28, 387	glossary-bookindex package	423
glossaries-stylemods package	glossary-hypernav package	101
461	glossary-long package	428
glossaries.sty package	glossary-longbooktabs package	428
50	glossary-longextra package	383–385
\GlossariesExtraWarning	glossary-tree package	384, 385, 435, 494
6,	\glossaryentrynumbers	
17, 19, 25, 32, 45–48, 66, 68, 80, 90, 73, 138, 162, 163, 167, 168	
131, 135, 146, 150, 153, 154, 160, 163,	\glossaryheader	
201–204, 206, 208, 214, 216, 245, 378, 386 145, 160, 163, 424–433, 435,	
\GlossariesExtraWarningNoLine	436, 438–441, 452, 454–457, 459, 464,	
..... 15, 19, 118, 127, 156	473, 474, 476–480, 482, 483, 485–491, 495	
\GlossariesWarning	\glossaryname	138
..... 71, 128, 135, 138, 142, 143, 162, 243	\glossarypostamble	145, 161, 163, 165
\GlossariesWarningNoLine	\glossarypreamble	145, 160, 163
133, 149	\glossarysection	145, 151, 153, 160, 163, 165
glossary styles:	\glossarytitle	
altlist 51, 137, 138, 145, 151, 153, 160, 162, 163	
425	\glossarytoctitle	
altlistgroup 51, 138, 140, 145, 151, 153, 160, 162, 163	
425	\glossentry	
altlisthypergroup 138, 162, 167, 168, 423–433, 435, 438,	
426	440, 452, 464, 473, 474, 476, 477, 479,	
alttree 383, 384, 437, 441, 442, 452, 470	480, 482, 483, 485, 487, 488, 490, 491, 495	
alttreegroup	\Glossentrydesc	493
453	\glossentrydesc	
alttreehypergroup 423, 424, 426–434, 437–440, 468, 494	
453	\Glossentryname	492
index	\glossentryname	158, 423–
435	433, 436, 438, 440, 452, 453, 461, 468, 494	
indexgroup	\glossentrynameother	159
436	\glossentrysymbol	428,
indexhypergroup	430, 432, 434, 438, 440, 442, 468, 492, 494	
436	\glossxtrsetpopts	217
inline	\GLS	116, 127, 171
434	\Gls	67, 116, 127, 171
list	\gls	47, 67, 69, 116, 127, 135, 150, 171
424, 425	\gls@assign@desc	49
listdotted	\gls@assign@field	14, 34, 95
423	\gls@checkseeallowed	63, 65, 66, 134
listdottedstyle	\gls@codepage	149
424	\gls@defdocnewglossaryentry . 63, 117, 124	
listgroup	\gls@defglossaryentry ... 49, 50, 64, 67, 68	
425	\gls@dotocitle	138, 162
listhypergroup		
425		
longragged-booktabs		
472		
mcolalttree		
458		
mcolalttreegroup		
458		
mcolalttreehypergroup		
459		
mcolalttreespannav		
459		
mcolindexgroup		
454		
mcolindexhypergroup		
455		
mcolindexspannav		
455		
mcoltreegroup		
455		
mcoltreehypergroup		
456		
mcoltreenonamegroup		
457		
mcoltreenonamehypergroup		
457		
mcoltreenonamespannav		
457		
mcoltreespannav		
456		
name-desc		
475		
sublistdotted		
424		
tree		
437		

<code>\gls@glossary</code>	59	<code>\glsaccesslong</code>	92, 93, 227, 235, 236, 247, 250, 252, 253, 256, 258, 260, 262–264, 267, 269, 271–279, 281, 284, 286–293, 295, 298, 300, 302, 304, 306, 308–320, 322, 323, 326, 327, 330, 333, 336, 338, 339, 342, 345, 350, 351
<code>\gls@grplabel</code>	102	<code>\Glsaccesslongpl</code>	94, 228, 239, 248, 258, 263, 264, 267, 274– 276, 284, 290–293, 301, 303, 311–318, 326–328, 336, 338, 339, 342, 345, 350, 351
<code>\gls@ifnotmeasuring</code>	12, 115	<code>\glsaccesslongpl</code>	93, 94, 228, 239, 240, 247, 250, 253, 256, 258, 260, 262–264, 267, 269, 271–279, 281, 284, 286–295, 298, 300, 302, 304, 306, 308–310, 312–320, 322, 323, 326, 327, 330, 333, 336, 338, 339, 342, 345, 350, 351
<code>\gls@level</code>	167	<code>\GLSaccessname</code>	85
<code>\gls@noidxglossary</code>	135	<code>\Glsaccessname</code>	85
<code>\gls@org@glossaryentryfield</code> ...	138, 162	<code>\glsaccessname</code>	85
<code>\gls@org@glossarysubentryfield</code>	138, 162	<code>\GLSaccessplural</code>	84
<code>\gls@org@TrackLangRequireDialectPrefix</code>	421	<code>\Glsaccessplural</code>	84
<code>\gls@save@numberlist</code>	71, 73	<code>\Glsaccessshort</code> 90, 234, 243, 250, 253, 256, 260, 262, 269, 271, 272, 278, 279, 281, 286, 287, 289, 295, 297, 298, 304, 306, 308, 309, 319, 320, 322, 323, 330, 333, 342, 347, 348
<code>\gls@set@xr@key</code>	60	<code>\glsaccessshort</code>	90, 91, 227, 228, 233, 234, 242, 247, 248, 250, 252, 253, 255, 256, 258–264, 267, 269, 271–273, 275, 276, 278, 281, 284, 285, 287–293, 295, 297, 298, 300, 302, 304, 306, 308–312, 314, 316, 317, 319, 320, 322, 326–328, 330, 333, 336, 338, 339, 342, 345, 347, 348, 351
<code>\gls@tmplen</code>	444–451, 453, 471, 472	<code>\Glsaccessshortpl</code> 91, 237, 243, 250, 253, 256, 260, 262, 269, 271, 272, 278, 279, 281, 286, 288, 289, 295, 297, 298, 304, 306, 308, 310, 320, 322, 323, 330, 333, 342, 347, 348, 351
<code>\gls@type</code>	135	<code>\glsaccessshortpl</code> 91, 92, 228, 237, 238, 243, 247, 248, 250, 252, 253, 256, 258–262, 264, 267, 269, 271–276, 278, 281, 284, 286–293, 295, 297, 298, 300–304, 306, 308–312, 314, 316–320, 322, 326, 328, 330, 333, 336, 339, 342, 345, 347, 348, 351
<code>\glsabbrvdefaultfont</code> ...	229, 247, 249, 252, 255, 258, 260, 263, 324, 335, 338, 349	<code>\GLSaccesssymbol</code>	87
<code>\glsabbrvmfont</code>	299–309, 311, 313, 315, 317, 319, 321–323	<code>\Glsaccesssymbol</code>	87, 215
<code>\glsabbrvfont</code>	105, 106, 131, 233, 234, 237, 238, 240, 242, 243, 246–255, 257–260, 263, 265, 267, 269, 270, 272, 274, 276, 278, 280, 283, 285, 287, 289, 291, 292, 294, 297, 300, 302, 304, 306, 308, 309, 311, 313, 315, 317, 319, 322, 326, 327, 329, 332–334, 336, 338, 341, 342, 344, 347, 350	<code>\glsaccesssymbol</code>	87, 215, 219, 220
<code>\glsabbrvhyphenfont</code>	335–337, 341–345, 347, 348		
<code>\glsabbrvonlyfont</code>	349–352		
<code>\glsabbrvscfont</code>	266–270, 272, 274, 276–280, 282		
<code>\glsabbrvsmfont</code>	283–289, 291, 292, 294, 296–298		
<code>\glsabbrvuserfont</code>	324–333		
<code>\GLSaccessdesc</code>	86		
<code>\Glsaccessdesc</code>	86, 202, 214		
<code>\glsaccessdesc</code>	86, 202, 219, 220		
<code>\GLSaccessdescplural</code>	86		
<code>\Glsaccessdescplural</code>	86		
<code>\glsaccessdescplural</code>	86		
<code>\GLSaccessfirst</code>	84		
<code>\Glsaccessfirst</code>	83		
<code>\glsaccessfirst</code>	83		
<code>\GLSaccessfirstplural</code>	85		
<code>\Glsaccessfirstplural</code>	85		
<code>\glsaccessfirstplural</code>	84		
<code>\Glsaccesslong</code>	92, 228, 236, 247, 258, 263, 264, 267, 273–276, 284, 290, 291, 293, 300, 302, 311–317, 326–328, 336, 338, 339, 342, 345, 350, 351		

<code>\GLSaccesssymbolplural</code>	87	<code>\glsdisplaynumberlist</code>	135, 142
<code>\Glsaccesssymbolplural</code>	87	<code>\glsdohyperlink</code>	101, 383
<code>\glsaccesssymbolplural</code>	87	<code>\glsdohypertarget</code>	104, 140
<code>\GLSaccesstext</code>	83	<code>\glsdoifexists</code>	18, 29, 41, 42, 49, 53, 55–59, 74, 75, 81, 90–94, 103, 115, 134, 142, 143, 157, 159, 230–240, 377
<code>\Glsaccesstext</code>	83	<code>\glsdoifexistsordo</code>	35, 36, 76
<code>\glsaccesstext</code>	82	<code>\glsdoifexistsorwarn</code>	18, 201, 202, 214, 215
<code>\glsacrshortcutstrue</code>	23, 24	<code>\glsdoifnoexists</code>	49
<code>\glsacspacemax</code>	132	<code>\glsdonohyperlink</code>	80, 103, 104
<code>\glsadd</code>	36, 63, 78, 82, 150	<code>\glsdosanitizesort</code>	136
<code>\glsadd options</code>		<code>\glsenableentrycount</code>	116, 118, 126
<code>theHvalue</code>	10	<code>\glsenableentryunitcount</code>	118, 127
<code>thevalue</code>	9, 10	<code>\glsentrycounter</code>	27, 147, 382
<code>\glsaddpostsetkeys</code>	10, 81	<code>\GlsEntryCounterLabelPrefix</code>	47
<code>\glsaddpresetkeys</code>	10, 81	<code>\glsentrycurrcount</code>	117, 118, 124
<code>\glsaddstoragekey</code>	61, 195, 379, 380	<code>\Glsentrydesc</code>	182, 193, 202
<code>\glsalttreechildpredesc</code>	442	<code>\glsentrydesc</code>	182, 193, 203
<code>\glsalttreepredesc</code>	441, 442	<code>\Glsentrydescplural</code>	183, 193
<code>\glsbackslash</code>	66	<code>\glsentrydescplural</code>	182, 183, 193
<code>\glscapscase</code>	75, 82–94, 96, 97, 230–242	<code>\Glsentryfirst</code>	122, 175, 180, 192
<code>\glscategory</code>		<code>\glsentryfirst</code>	121, 175, 180, 192, 370
... 73, 82, 97, 105, 106, 196–198, 201–		<code>\Glsentryfirstplural</code>	122, 175, 180, 192
208, 214, 215, 217–219, 230–234, 237, 238		<code>\glsentryfirstplural</code>	
<code>\glscategorylabel</code> 122, 175, 180, 181, 192, 371	
... 97, 186, 189, 190, 222, 224–226, 244,		<code>\glsentryfmt</code>	50–53, 131
247–250, 252, 254–258, 260, 265–270,		<code>\Glsentryfull</code>	129
272, 277, 279–288, 290, 294, 296–298,		<code>\glsentryfull</code>	129
300, 301, 303, 305, 307, 309, 311, 313,		<code>\Glsentryfullpl</code>	129
314, 318, 320, 321, 323, 325, 326, 328,		<code>\glsentryfullpl</code>	129
329, 331–333, 335, 336, 339, 341, 343–351		<code>\glsentryitem</code>	158, 159, 423–
<code>\glsclosebrace</code>	59, 151, 152	433, 435, 438, 440, 452, 464, 468, 492, 494	
<code>\glscounter</code>	9, 19	<code>\Glsentrylong</code> ...	106, 107, 122, 175, 184, 194
<code>\glscurrententrylabel</code>		<code>\glsentrylong</code>	106, 107,
... 71, 72, 138, 148, 158–164, 217		121, 175, 184, 194, 255, 257, 280, 282,	
<code>\glscurrentfieldvalue</code>		297, 299, 321, 323, 329, 331, 346, 372, 373	
35, 36, 38–41, 43–45, 56, 57, 324, 378, 379		<code>\Glsentrylongpl</code> ...	106, 107, 122, 185, 194
<code>\glscustomtext</code> ..	74–76, 90–94, 230–240, 242	<code>\glsentrylongpl</code>	
<code>\glsdefaultshortaccess</code>	188, 189	... 106, 107, 122, 185, 194, 372, 373	
<code>\glsdefaulttype</code>		<code>\Glsentrylongplural</code>	175
... 6, 21, 47, 48, 137, 150, 160–162, 165		<code>\glsentrylongplural</code>	175
<code>\glsdescriptionaccessdisplay</code> ..	182, 202	<code>\Glsentryname</code>	178, 191, 203, 205, 206
<code>\glsdescriptionpluralaccessdisplay</code>		<code>\glsentryname</code>	157, 158,
... 182, 183		178, 191, 192, 210, 367, 368, 445–451, 467	
<code>\glsdescwidth</code>	426–433, 470–472	<code>\glsentrynumberlist</code> ...	136, 143, 449–451
<code>\glsdetoklabel</code>	8, 9,	<code>\glsentrypdfsymbol</code>	214
33, 37–43, 47–50, 53–55, 58, 62, 64–66,		<code>\Glsentryplural</code>	179, 192
78, 81, 99, 103, 117, 118, 123–127, 135,		<code>\glsentryplural</code>	179, 192, 369, 370
138, 142, 143, 158, 159, 162, 166, 167,		<code>\glsentryprevcount</code>	117, 119, 125
170–172, 201, 204, 205, 208, 209, 446, 447			
<code>\glsdisp</code>	389		

<code>\glsentryprevmaxcount</code>	125	<code>\glsfirstabbrvscfont</code>	266–282
<code>\glsentryprevtotalcount</code>	125	<code>\glsfirstabbrvsmfont</code>	283–298
<code>\Glsentryshort</code>	105, 107, 183, 194	<code>\glsfirstabbrvuserfont</code> .	325–330, 332–334
<code>\glsentryshort</code>	105–107, 132, 183, 193, 194, 327, 328, 340, 366, 367, 373	<code>\glsfirstaccessdisplay</code>	180
<code>\Glsentryshortpl</code>	106, 107, 184, 194	<code>\glsfirstlongdefaultfont</code>	247, 250, 258, 260, 263, 266–277, 283– 294, 300, 301, 304, 305, 308–312, 315, 316
<code>\glsentryshortpl</code>	106, 107, 184, 194, 366, 367, 373	<code>\glsfirstlongemfont</code>	302, 303, 306, 307, 313, 314, 317, 318
<code>\Glsentrysymbol</code>	181, 193	<code>\glsfirstlongfont</code>	227, 228, 247–252, 255, 258, 260, 262–265, 267, 269, 271, 272, 274, 276, 278, 280, 283, 285, 287, 289, 291, 292, 294, 297, 300, 302, 304, 306, 308, 309, 311, 313, 315, 317, 319, 322, 326, 327, 330, 333, 336, 338, 342, 345, 347, 350
<code>\glsentrysymbol</code>	181, 193, 214, 448–450	<code>\glsfirstlongfootnotefont</code>	252–257, 277–282, 294–299, 319–323
<code>\Glsentrysymbolplural</code>	182, 193	<code>\glsfirstlonghyphenfont</code>	335–347
<code>\glsentrysymbolplural</code>	181, 182, 193	<code>\glsfirstlongonlyfont</code>	350, 351
<code>\Glsentrytext</code>	179, 192	<code>\glsfirstlonguserfont</code>	325–334
<code>\glsentrytext</code> ...	103, 178, 179, 192, 368, 369	<code>\GLSfirstplural</code>	361, 362
<code>\glsentrytype</code>	158, 227	<code>\Glsfirstplural</code>	362
<code>\Glsentryuseri</code>	88	<code>\glsfirstplural</code>	361, 362
<code>\glsentryuseri</code>	88	<code>\glsfirstpluralaccessdisplay</code> ..	180, 181
<code>\Glsentryuserii</code>	88	<code>\GLSfmtname</code>	57
<code>\glsentryuserii</code>	88	<code>\Glsfmtname</code>	56, 57
<code>\Glsentryuseriii</code>	88	<code>\glsfmtname</code>	56, 57
<code>\glsentryuseriii</code>	88	<code>\GLSfmtttext</code>	57
<code>\Glsentryuseriv</code>	89	<code>\Glsfmtttext</code>	56, 57
<code>\glsentryuseriv</code>	89	<code>\glsfmtttext</code>	56, 57
<code>\Glsentryuserv</code>	89	<code>\glsforeachincategory</code>	243
<code>\glsentryuserv</code>	89	<code>\glsgenentryfmt</code>	73
<code>\Glsentryuservi</code>	89	<code>\glsgetattribute</code>	80, 102, 119, 123– 125, 148, 172, 176, 201–206, 208, 210, 214
<code>\glsentryuservi</code>	89	<code>\glsgetcategoryattribute</code>	196
<code>\glsextrapostnamehook</code>	207	<code>\glsgetgrouptitle</code>	425, 426
<code>\glsfieldfetch</code>	103	<code>\glsgetwidestname</code>	442
<code>\glsfieldxdef</code>	200, 201	<code>\glsgroupheading</code>	166, 424–433, 435, 436, 438–441, 452–459, 465, 473, 474, 476, 477, 479, 480, 482, 483, 485, 487, 488, 490, 491, 495
<code>\glsFindWidestLevelTwo</code>	386	<code>\glsgroupskip</code>	166, 424, 427–434, 436, 439, 440, 453, 465, 473, 475–479, 481, 482, 484, 485, 487, 488, 490, 492, 496
<code>\glsFindWidestTopLevelName</code>	385, 386	<code>\glshasattribute</code>	79, 102, 118, 119, 123, 125, 127, 148, 171, 176, 201–206, 208, 210, 214, 247–249, 251, 252, 254, 255, 257, 259, 261, 262, 264,
<code>\glsfindwidesttoplevelname</code>	444		
<code>\GLSfirst</code>	360, 361		
<code>\Glsfirst</code>	361		
<code>\glsfirst</code>	361		
<code>\glsfirstabbrvdefaultfont</code>	229, 247, 250, 252, 255, 258, 260, 263, 338		
<code>\glsfirstabbrvsmfont</code>	300–323		
<code>\glsfirstabbrvfont</code>	131, 227, 228, 247–256, 258–264, 267, 269, 271, 272, 274, 276, 278, 280, 283, 285, 287, 289, 291, 292, 294, 297, 300, 302, 304, 306, 308, 309, 311, 313, 315, 317, 319, 322, 326, 327, 329, 333, 336, 338, 339, 342, 344, 347, 350		
<code>\glsfirstabbrvhyphenfont</code>	335–337, 340, 342, 344–348		
<code>\glsfirstabbrvonlyfont</code>	350, 351		

265, 267–270, 277, 279, 280, 282–286, 294, 296, 297, 299–307, 315, 318, 319, 321–323, 325, 327–329, 331–334, 336– 338, 340, 341, 343–345, 347, 349, 350, 352	\glslinkpresetkeys 10, 79, 172
\glslinkcategoryattribute 196	\glslinkvar 100, 101
\glslinkhex .. 392–397, 400–405, 408–410, 412–421	\glslistchildpostlocation 424
\glslinkhyperlink 103, 379	\glslistchildprelocation 424, 425
\glslinkhypernavsep 145	\glslistdesc 424, 425
\glslinkhypernumber 148, 209, 380–382	\glslistdottedwidth 423
\glslinkifattribute . 76, 77, 83, 98, 100, 109, 176, 199, 202–205, 208, 217, 220, 356–366	\glslistgroupheaderfmt 425, 426
\glslinkifcategory 198	\glslistgroupskip 424
\glslinkifcategoryattribute 97, 186, 189, 190, 197, 198, 225, 226	\glslistnavigationitem 425, 426
\glslinkifnotregular 82	\glslistprelocation 424, 425
\glslinkifnotregularcategory 198	\glslistlocalunset 75, 172
\glslinkifplural 75, 82, 84–87, 90–94, 220, 230–241	\glslongaccessdisplay 184
\glslinkifregular 73, 82, 121, 122, 175	\glslongdefaultfont . 229, 247, 250, 251, 258, 260, 263, 267, 269, 271, 272, 274, 276, 283, 285, 287, 289, 291–293, 300, 304, 308, 309, 311, 312, 315, 325, 335, 349
\glslinkifregularcategory 198	\glslongemfont .. 299, 302, 305, 306, 313, 317
\glslinkifusetranslator 51	\glslongextraDescAlign 472–483, 485, 486, 488–490
\glslinkignore 72	\glslongextraDescFmt 469, 473, 474, 476, 477, 479, 480, 482, 484, 485, 487, 488, 490
\glslinkinlinedescformat 434	\glslongextraDescNameHeader 476
\glslinkinlinesubdescformat 434	\glslongextraDescNameTabularFooter 475
\glslinkinsert 75, 82, 90–94, 230–242, 334, 341, 343, 347, 349	\glslongextraDescNameTabularHeader 475, 476
\glskeylisttok 129, 224, 227	\glslongextraDescSymNameHeader 488
\glslabel 8, 9, 35, 36, 53, 73, 76–80, 97–99, 102, 103, 132, 172, 176, 218–220, 240–242, 255, 257, 280, 282, 297, 299, 321, 323, 327–329, 331, 341, 343, 347, 349	\glslongextraDescSymNameTabularFooter 487, 488
\glslabeltok 129, 224, 227, 247–249, 251– 255, 257–270, 272, 274, 277–280, 282– 287, 289, 291, 294, 296, 297, 299–309, 311, 313, 315, 317–323, 325–334, 336– 338, 340, 341, 343–345, 347, 349, 350, 352	\glslongextraDescSymNameTabularHeader 487, 488
\glsletentryfield 210	\glslongextraGroupHeading 473, 474, 476, 477, 479, 480, 482, 483, 485, 487, 488, 490
\glslink 129, 389	\glslongextraHeaderFmt . 470, 471, 473, 475, 476, 478, 479, 481–484, 486, 487, 489
\glslink options	\glslongextraLocationAlign 474, 477, 480, 483, 486, 489, 490
counter 10	\glslongextraLocationDescNameHeader 477
format 209	\glslongextraLocationDescNameTabularFooter 476, 477
hyper 352	\glslongextraLocationDescNameTabularHeader 476, 477
hyperoutside 77	\glslongextraLocationDescSymNameHeader 490
noindex 9, 98, 352	\glslongextraLocationDescSymNameTabularFooter 489
textformat 79	\glslongextraLocationDescSymNameTabularHeader 489
theHvalue 79	\glslongextraLocationFmt 474, 477, 480, 484, 487, 490
thevalue 79, 168	
wrgloss 8, 76	
\glslinkcheckfirsthyperhook 97	
\glslinkpostsetkeys 10, 79, 172	

<code>\glslongextraLocationSymDescNameHeader</code>	<code>\glslongextraSubNameFmt</code> 473, 475–477, 479, 480, 482, 484, 485, 487, 488, 490
..... 487	
<code>\glslongextraLocationSymDescNameTabularFooter</code>	<code>\glslongextraSubSymbolFmt</code> 479, 481, 482, 484, 485, 487, 488, 490
..... 485, 486	
<code>\glslongextraLocationSymDescNameTabularHeader</code>	<code>\glslongextraSymbolAlign</code> 478–486, 488–490
..... 485, 486	
<code>\glslongextraLocSetDescWidth</code> .. 474, 477	<code>\glslongextraSymbolFmt</code> 469, 479, 480, 482, 484, 485, 487, 488, 490
<code>\glslongextraNameAlign</code> 472–483, 485, 486, 488–490	<code>\glslongextraSymDescNameHeader</code> 485
<code>\glslongextraNameDescHeader</code> 473	<code>\glslongextraSymDescNameTabularFooter</code> 484, 485
<code>\glslongextraNameDescLocationHeader</code> 474	<code>\glslongextraSymDescNameTabularHeader</code> 484, 485
<code>\glslongextraNameDescLocationTabularFooter</code> 473, 474	<code>\glslongextraSymLocSetDescWidth</code> ...
<code>\glslongextraNameDescLocationTabularHeader</code> 473, 474	<code>\glslongextraSymSetDescWidth</code> 480, 483, 486, 489
<code>\glslongextraNameDescSymHeader</code> 479 472, 478, 481, 482, 484, 485, 488
<code>\glslongextraNameDescSymLocationHeader</code> 480	<code>\glslongextraTabularVAlign</code> 472, 474, 475,
<code>\glslongextraNameDescSymLocationTabularFooter</code> 479, 480	477, 478, 480, 481, 483, 484, 486, 488, 489
<code>\glslongextraNameDescSymLocationTabularHeader</code> 479, 480	<code>\glslongextraUpdateWidest</code> 383–385
<code>\glslongextraNameDescSymTabularFooter</code> 478	<code>\glslongextraUpdateWidestChild</code> 383–386
<code>\glslongextraNameDescSymTabularHeader</code> 478	<code>\GlsLongExtraUseTabularfalse</code> 472
<code>\glslongextraNameDescTabularFooter</code> 470, 473	<code>\glslongfont</code> 106, 107, 229, 235, 236, 239, 240, 247,
<code>\glslongextraNameDescTabularHeader</code> 470, 473	248, 250, 252, 253, 255, 258–260, 262,
<code>\glslongextraNameFmt</code> ... 473, 474, 476, 477, 479, 480, 482, 484, 485, 487, 488, 490	263, 265, 267, 269, 271, 272, 274, 276,
<code>\glslongextraNameSymDescHeader</code> 482	278, 280, 283, 285, 287, 289, 291, 292,
<code>\glslongextraNameSymDescLocationHeader</code> 483	294, 297, 300, 302, 304, 306, 308, 309,
<code>\glslongextraNameSymDescLocationTabularFooter</code> 482, 483	311, 313, 315, 317, 319, 322, 326, 327,
<code>\glslongextraNameSymDescLocationTabularHeader</code> 482, 483	330, 333, 336, 338, 342, 345, 347, 350, 351
<code>\glslongextraNameSymDescTabularFooter</code> 481	<code>\glslongfootnotefont</code> 251, 252, 255, 278, 280, 294, 297, 319, 322
<code>\glslongextraNameSymDescTabularHeader</code> 481, 482	<code>\glslonghyphenfont</code> 335, 336, 338, 339, 341, 342, 344, 345, 347
<code>\glslongextraSetDescWidth</code> 471–473, 475, 476	<code>\glslongonlyfont</code> 349, 350
<code>\glslongextraSubDescFmt</code> 473, 475–477, 479, 481, 482, 484, 485, 487, 488, 490	<code>\glslongpltok</code> 226, 227, 247–249, 251, 252,
<code>\glslongextraSubLocationFmt</code> 475, 477, 481, 484, 487, 490	254, 263, 265, 266, 268, 270, 274, 277,
	279, 283–286, 291, 294, 296, 300–307,
	311, 313, 317, 319, 321, 325, 326, 328–
	332, 334, 336–339, 341, 343–345, 350, 351
	<code>\glslongpluralaccessdisplay</code> 185
	<code>\glslongtok</code> 129, 224, 225,
	227, 247–255, 258, 259, 262, 263, 265–
	268, 270, 274, 277, 279, 280, 283–287,
	291, 294, 296, 300–307, 311, 313, 317,
	319–321, 325, 326, 328, 329, 331, 332,
	334, 336–339, 341, 343–345, 347, 350, 351
	<code>\glslonguserfont</code> 325–328, 330, 333
	<code>\glsmlcols</code> 455, 456, 458, 459

<code>\GLSname</code>	358	<code>\glssseeformat</code>	55, 58, 64, 135, 142, 143
<code>\Glsname</code>	358	<code>\glssseelist</code>	58
<code>\glsname</code>	358	<code>\glsssetabbrvfmt</code> ...	73, 82, 105, 106, 201–206, 208, 214, 215, 230–234, 237, 238, 240
<code>\glsnameaccessdisplay</code> ..	178, 203, 204, 206	<code>\glsssetattribute</code>	247, 249, 251, 252, 254, 255, 257–270, 272, 274, 277–280, 282–287, 289, 291, 294, 296, 297, 299–309, 311, 313, 315, 317–319, 321–323, 325, 327, 329, 331–334, 336–338, 340, 341, 343–345, 347, 349, 350, 352
<code>\glsnamefont</code>	203–206, 208, 471	<code>\glsssetcategoryattribute</code>	116, 128, 132, 171, 177, 190, 191, 196, 197, 199, 200, 215
<code>\glsnavhyperlink</code>	145	<code>\glsssetnoexpandfield</code>	14
<code>\glsnavhyperlinkname</code>	102	<code>\glsssettoctitle</code>	138, 162
<code>\glsnavhypertarget</code>	425, 426, 436, 439, 441, 454–460	<code>\glsssetwidest</code>	384, 385
<code>\glsnavigation</code>	425, 426, 436, 439, 441, 454–459	<code>\glssshortaccessdisplay</code>	183
<code>\glsnextpages</code>	138, 162	<code>\glssshortaccsupp</code>	190
<code>\glsnoidxdisplayloc</code>	142, 143, 380	<code>\glssshortpltok</code>	226, 227, 247–249, 251, 252, 254–258, 260, 266, 268, 270, 272, 277, 279, 280, 282–288, 294, 296, 298, 300–307, 309, 319–321, 323, 325, 326, 328, 329, 331, 332, 334, 336, 337, 341, 343–345, 347, 348, 350, 352
<code>\glsnoidxdisplayloclisthandler</code>	142	<code>\glssshortpluralaccessdisplay</code>	184
<code>\glsnoidxloclist</code>	143, 167	<code>\glssshorttok</code>	129, 224, 226, 227, 246–260, 265, 266, 268, 270, 272, 274, 277, 279, 280, 282–288, 291, 294, 296, 298, 300–307, 309, 311, 313, 319–321, 323, 325, 326, 328–332, 334–337, 339, 341, 343–345, 347–351
<code>\glsnoidxnumberlistloophandler</code>	142	<code>\glssshowtargetfont</code>	29
<code>\glsnonnextpages</code>	138, 162	<code>\glssshowtargetouter</code>	28, 29
<code>\glsnonnumberlistfalse</code>	71	<code>\glssubentryitem</code>	158, 423–434, 436, 438, 440, 452, 465, 469
<code>\glsnonnumberlisttrue</code>	71	<code>\glssymbolaccessdisplay</code>	181
<code>\glsnopostdotfalse</code>	139	<code>\glssymbolpluralaccessdisplay</code> .	181, 182
<code>\glsnopostdottrue</code>	139	<code>\glstarget</code> ...	158, 159, 423–434, 436, 438, 440, 452, 453, 464, 465, 468, 469, 492, 494
<code>\glsnumberlistloop</code>	136	<code>\GLStext</code>	359
<code>\glsnumlistlastsep</code>	142, 377	<code>\Glstext</code>	359
<code>\glsnumlistsep</code>	142, 377	<code>\glstext</code>	359
<code>\glsopenbrace</code>	59, 151, 152	<code>\glstextaccessdisplay</code>	178, 179
<code>\glsorder</code>	134, 156	<code>\glstextformat</code>	76, 80
<code>\glspagelistwidth</code> 427, 429, 431, 433, 470, 472		<code>\glstextup</code>	266
<code>\glspar</code>	165, 492	<code>\glstopic@postchildren</code>	495, 496
<code>\glspatchLToutput</code>	473, 474, 476–478, 480, 482, 483, 485, 486, 488, 489	<code>\glstopic@prechildren</code>	491, 495, 496
<code>\glspdfdfmtfull</code>	374	<code>\glstopic@prevlevel</code>	491, 495, 496
<code>\glspdfdfmtfullpl</code>	374, 375	<code>\glstopicAssignSubIndent</code>	492, 496
<code>\glspenaltygroupskip</code> ...	473, 475, 476, 478, 479, 481, 482, 484, 485, 487, 488, 490	<code>\glstopicAssignWidest</code>	493
<code>\GLSpl</code>	116, 128, 171		
<code>\Glspl</code>	68, 116, 127, 171		
<code>\glspl</code>	68, 116, 127, 171		
<code>\GLSplural</code>	360		
<code>\Glsplural</code>	360		
<code>\glsplural</code>	360		
<code>\glspluralaccessdisplay</code>	179		
<code>\glspluralsuffix</code>	155, 225, 229		
<code>\glspostdescription</code>	20, 139, 217, 423, 424, 426–434, 437–440, 468, 493, 494		
<code>\glspostinline</code>	434		
<code>\glspostlinkhook</code> 74, 76, 90–94, 108, 230–240			
<code>\glsprestandardsort</code>	136		
<code>\glsresetentrylist</code>	145, 160, 163		
<code>\glsssee</code>	59, 61, 62		

<code>\glstopicCols</code>	496	<code>\glstreepredesc</code>	437, 439
<code>\glstopicColsEnv</code>	496	<code>\glstreePreHeader</code> ..	436, 439, 441, 453–459
<code>\glstopicDesc</code>	492	<code>\glstreeprelocation</code>	435, 437, 440
<code>\glstopicGroupHeading</code>	491, 495	<code>\glstreesubitem</code>	435, 465
<code>\glstopicInit</code>	491, 495	<code>\glstreesubsubitem</code>	435, 465
<code>\glstopicItem</code>	491, 496	<code>\glstreesymbol</code>	436, 438
<code>\glstopicLoc</code>	492	<code>\GlsTrLetField</code>	42
<code>\glstopicMarker</code>	492	<code>\glstype</code>	75, 78, 90–94, 172, 230–240
<code>\glstopicMidSkip</code>	492	<code>\glunset</code>	63, 75, 119, 120, 172
<code>\glstopicParIndent</code>	491, 496	<code>\glupdatewidest</code>	383, 384
<code>\glstopicPostSkip</code>	492	<code>\gluserdescription</code>	325, 326, 329, 332
<code>\glstopicPreSkip</code>	492	<code>\glswrite</code>	59, 134
<code>\glstopicSubIndent</code>	493	<code>\glswriteentry</code>	8, 9
<code>\glstopicSubItem</code>	492, 496	<code>\Glsxtr</code>	68
<code>\glstopicSubItemBox</code>	494	<code>\glxtr</code>	68
<code>\glstopicsubitemhangindent</code>	493	<code>\glxtr@do@wrglossary</code>	8, 10, 13, 16
<code>\glstopicSubItemParIndent</code>	493	<code>\glxtr@addloclistfield</code>	15, 17
<code>\glstopicSubItemSep</code>	494	<code>\glxtr@addunused</code>	62
<code>\glstopicSubLoc</code>	494	<code>\glxtr@applyabbrvfmt</code>	240
<code>\glstopicSubNameFont</code>	494	<code>\glxtr@applyabbrvstyle</code>	225, 244
<code>\glstopicSubPreLocSep</code>	494	<code>\glxtr@beginbookindex</code>	463
<code>\glstopicTitle</code>	492	<code>\glxtr@counterrecord</code>	157
<code>\glstopicTitleFont</code>	492	<code>\glxtr@dblfloat</code>	19
<code>\glstopicwidest</code>	493–495	<code>\glxtr@do@alsoindex@wrglossary</code>	17
<code>\glstreechilddesc</code>	438	<code>\glxtr@do@autoadd</code>	9, 78, 79
<code>\glstreeChildDescLoc</code>	436, 439	<code>\glxtr@doooption</code>	5, 19, 20, 27, 28, 30
<code>\glstreechildpredesc</code>	437	<code>\glxtr@endbookindex</code>	464
<code>\glstreechildprelocation</code>	438, 440	<code>\glxtr@fields</code>	155
<code>\glstreechildsymbol</code>	436, 439	<code>\glxtr@float</code>	19
<code>\glstreedefaultnamefmt</code>	435	<code>\glxtr@grptitle</code> ...	436, 439, 441, 453–460
<code>\glstreedesc</code>	437	<code>\glxtr@headentry@p</code>	109, 110
<code>\glstreeDescLoc</code>	436, 438, 442	<code>\glxtr@hyperoutsidefalse</code>	77
<code>\glstreegroupheaderfmt</code>		<code>\glxtr@hyperoutsidettrue</code>	77
.....	436, 439, 441, 453–460, 462	<code>\glxtr@ifnextpunc</code>	221
<code>\glstreegroupheaderskip</code>		<code>\glxtr@ifpunctoken</code>	221
.....	436, 437, 439, 441, 453–460	<code>\glxtr@inc@linkcount</code>	79, 177
<code>\glstreegroupskip</code> ..	435, 436, 439, 440, 453	<code>\glxtr@inc@wrglossaryctr</code>	8, 9, 27, 31
<code>\glstreeindent</code>	438, 440, 451–453	<code>\glxtr@indexonly@saveentrycounter</code>	
<code>\glstreeitem</code>	435, 455, 464, 465	16, 17, 31
<code>\glstreenamebox</code>	452, 453	<code>\glxtr@keylist</code>	67, 68
<code>\glstreenamefmt</code>		<code>\glxtr@label</code>	467
.....	434, 436, 438, 440, 442, 445–453	<code>\glxtr@langtag</code>	155
<code>\glstreenavigationfmt</code>	436, 439, 441, 454–459	<code>\glxtr@linkprefix</code>	155, 156
<code>\glstreeNoDescSymbolPreLocation</code>	437, 438	<code>\glxtr@loaddialect</code>	376, 421
<code>\glstreenonamechilddesc</code>	440	<code>\glxtr@locationhypertext</code>	382
<code>\glstreenonameChildDescLoc</code>	440	<code>\glxtr@makeglossaries</code>	134
<code>\glstreenonamedesc</code>	440	<code>\glxtr@newabbreviation</code>	131, 224
<code>\glstreenonameDescLoc</code>	440	<code>\glxtr@next</code>	221, 222
<code>\glstreenonamesymbol</code>	440	<code>\glxtr@org@do@wrglossary</code>	31

\glxtr@org@dohyperlink	102	\glxtrAccSuppAbbrSetTextShortAttrs	
\glxtr@org@getgrouptitle	144	248,
\glxtr@org@newignoredglossary	50	250, 267, 269, 284, 286, 301, 303, 305,	
\glxtr@org@makenoidxglossaries	64	307, 328, 331, 333, 336, 343, 345, 348, 351	
\glxtr@pluralsuffixes	155	\glxtractivatenopost	138
\glxtr@printgloss@checkexists	138	\glxtraddallcrossrefs	62
\glxtr@printgloss@groupstrue	141	\glxtral alias	99
\glxtr@process	160, 161, 163, 164	\glxtrAltTreeIndent	442
\glxtr@provideignoredglossary	52	\glxtral treeInit	452, 458, 459
\glxtr@punctlist	221	\glxtrAltTreePar	442
\glxtr@record	11, 12, 155	\glxtrAltTreeSetHangIndent ...	442, 452
\glxtr@record@nameref	12, 155	\glxtrAltTreeSetSubHangIndent ...	453
\glxtr@record@nr	16	\glxtral treeSubSymbolDescLocation	453
\glxtr@recordsee	13	\glxtral treeSymbolDescLocation ..	
\glxtr@renewcommand	378	442, 452
\glxtr@resource	153, 155	\glxtrassignactualsetup	187
\glxtr@s@newignoredglossary	50	\glxtrassignfieldfont	82–89
\glxtr@s@provideignoredglossary ...	52	\glxtrautoindex	210
\glxtr@saveentrycounter	8, 10, 13, 99	\glxtrautoindexassignsort	210
\glxtr@setaccessdisplay	208	\glxtrautoindexentry	210
\glxtr@setbookindexmark	466	\glxtrautoindexesc	210
\glxtr@setup@record	15–17, 31, 32	\glxtrbibaddress	379
\glxtr@shortcutsval	155, 156	\glxtrbibauthor	379
\glxtr@texencoding	155, 156	\glxtrbibbooktitle	379
\glxtr@undefaction@nr	7	\glxtrbibchapter	379
\glxtr@undefaction@val	7	\glxtrbibedition	379
\glxtr@usese	55	\glxtrbibhowpublished	379
\glxtr@warnonexistsordo	7, 15–17, 54	\glxtrbibinstitution	379
\glxtr@writefields	153	\glxtrbibjournal	380
\glxtrabbreviationfont	73	\glxtrbibmonth	380
\glxtrabbrvfootnote ...	252–255, 257,	\glxtrbibnote	380
277–280, 282, 294–297, 299, 319–321, 323		\glxtrbibnumber	380
\glxtrabbrvpluralsuffix ...	155, 188,	\glxtrbiborganization	380
229, 247, 249, 252, 255, 258, 260, 263,		\glxtrbibpages	380
266, 283, 299, 325, 335, 338, 341, 347, 349		\glxtrbibpublisher	380
\glxtrabbrvtype	21, 227	\glxtrbibschoo	380
\glxtrAccSuppAbbrSetFirstLongAttrs		\glxtrbibseries	380
.....	247, 249,	\glxtrbibtitle	380
266, 268, 283, 285, 300, 301, 303, 305,		\glxtrbibtype	380
325, 326, 329, 332, 335, 341, 344, 346, 350		\glxtrbibvolume	380
\glxtrAccSuppAbbrSetNameLongAttrs		\glxtrbookindexatendgroup	464
.....	254,	\glxtrbookindexatsubendgroup	465
256, 272, 279, 281, 287, 296, 298, 320, 323		\glxtrbookindexatsubsubendgroup ..	465
\glxtrAccSuppAbbrSetNameShortAttrs		\glxtrbookindexbetween	464
.....	265, 290, 311, 313, 314, 339	\glxtrbookindexbookmark	466
\glxtrAccSuppAbbrSetNoLongAttrs ..		\glxtrbookindexbookmarkprefix	466
.....	252, 254, 258, 260, 270,	\glxtrbookindexcols	463, 464
277, 280, 288, 294, 296, 307, 309, 318, 321		\glxtrbookindexcolspread	463
		\glxtrbookindexfirstmarkfmt	467

\glxstrbookindexformatheader	466	\glxstremsuffix	300, 302, 304, 306, 308, 309, 311, 313, 315, 317, 319, 322
\glxstrbookindexgroupskip	466	\GlsXtrEnableEntryCounting	128
\glxstrbookindexlastmarkfmt	467	\GlsXtrEnableEntryUnitCounting	116
\glxstrbookindexlocation	462, 464	\GlsXtrEnableOnTheFly	66, 69
\glxstrbookindexmulticolsekv ..	463, 464	\glxstrendfor	38
\glxstrbookindexname	461, 464	\glxstrfieldlistgadd	157
\glxstrbookindexparentchildsep	462–464	\glxstrfieldtitlecase ..	202, 203, 205, 208
\glxstrbookindexparentsubchildsep .	463–465	\glxstrfieldtitlecasecs	201
\glxstrbookindexprelocation ...	461, 464	\glxstrfieldxifinlist	165
\glxstrbookindexsubbetween	465	\glxstrfirstscfont	266
\glxstrbookindexsublocation	465	\glxstrfirstsmfont	283
\glxstrbookindexsubname	465	\GlsXtrFmtDefaultOptions	35, 36
\glxstrbookindexsubprelocation ...	465	\glxstrfmtdisplay	35, 36
\glxstrbookindexsubsubbetween	465	\glxstrfmtexternalnameref	382
\glxstrbookindexthepage	466, 467	\GlsXtrFmtField	35, 36
\glxstrcat	67, 68	\glxstrfmtinternalnameref	381, 382
\glxstrchecknohyperfirst	83–85	\glxstrfootnotedescname	254, 256, 279, 282, 296, 298, 320, 323
\glxstrcombiningdiacriticIIrules .	393	\glxstrfootnotedescsort	254, 256, 279, 282, 296, 298, 320, 323
\glxstrcombiningdiacriticIIrules ..	393	\glxstrfootnotename	252, 255, 277, 280, 294, 296, 319, 321
\glxstrcombiningdiacriticIrules ...	393	\GlsXtrForeignTextField	45
\glxstrcombiningdiacriticIVrules ..	393	\GlsXtrFormatLocationList	71, 73, 449–451
\glxstrComputeTreeIndent	452, 453	\GLSxtrfull	22, 95, 364, 365
\glxstrComputeTreeSubIndent	453	\Glsxtrfull	22, 94, 365
\glxstrcounterprefix	147	\glxstrfull	21, 22, 94, 364
\glxstrcurrencyrules	395	\Glsxtrfullformat	228, 242, 244, 245, 247, 250, 253, 256, 259, 261, 264, 267, 269, 271, 273, 275, 277, 278, 281, 284, 286, 288, 289, 292, 293, 295, 297, 300, 302, 304, 306, 308, 310, 312, 314, 316, 318, 319, 322, 326, 327, 330, 333, 336, 339, 342, 345, 348, 350
\glxstrcurrentgrptitle	466	\glxstrfullformat	228, 242, 244, 245, 247, 250, 252, 255, 259, 261, 264, 267, 269, 271, 273, 275, 277, 278, 281, 284, 285, 288, 289, 292, 293, 295, 297, 300, 302, 304, 306, 308, 310, 312, 314, 316, 318, 319, 322, 326, 327, 330, 333, 336, 339, 342, 345, 347, 350
\GlsXtrDefaultResourceOptions	153	\GLSxtrfullpl	22, 95, 365, 366
\Glsxtrdefaultsubsequentfmt ...	243, 244	\Glsxtrfullpl	22, 95, 366
\glxtrdefaultsubsequentfmt ...	242, 244	\glxstrfullpl	21, 22, 95, 365
\Glsxtrdefaultsubsequentplfmt .	243, 244	\Glsxtrfullplformat	228, 242, 244, 245, 248, 250, 253, 256, 259, 261, 264, 267, 269, 271, 273, 275, 277, 278, 281, 284, 286, 288, 289, 292,
\glxtrdefaultsubsequentplfmt .	243, 244		
\GlsXtrDefineAbbreviationShortcuts	23, 24		
\GlsXtrDefineAcShortcuts	24		
\GlsXtrDefineOtherShortcuts	24		
\glxtrdetoklocation	171		
\glxtrdiscardperiod	218		
\glxtrdisplayendloc	146		
\glxtrdisplayendloohook	146		
\glxtrdisplayingleloc	146		
\glxtrdisplaystartloc	146		
\glxtrdoautoindexname	100, 207		
\glxtrdohyperlink	102, 104		
\glxtrdopostpunc	255, 257, 280, 282, 297, 299, 321, 323		
\glxtrdowrglossaryhook	100		
\GlsXtrDualField	379		

294, 295, 297, 301, 303, 304, 306, 309, 310, 312, 314, 316, 318, 320, 322, 326, 327, 330, 333, 336, 339, 342, 345, 348, 350	\glxstrifemptyglossary . 145, 151, 160, 163
\glxstrfullplformat	\GlsXtrIfFieldEqNum 158
.... 242, 244, 245, 247, 250, 252, 255, 259, 261, 264, 267, 269, 271, 273, 275, 277, 278, 281, 284, 286, 288, 289, 292, 293, 295, 297, 300, 302, 304, 306, 308, 310, 312, 314, 316, 318, 319, 322, 326, 327, 330, 333, 336, 339, 342, 345, 348, 350	\GlsXtrIfFieldNonZero 378
\glxstrfullsep	\glxstrifhasfield .. 45, 56, 57, 378, 379, 461
.... 227, 228, 247–251, 253, 256, 258– 260, 262–264, 266–276, 278, 279, 281, 283–293, 295, 298, 300–312, 314, 316– 318, 320, 322–324, 335–340, 344–346, 351	\glxstrifhyphenstart 335, 337, 340, 344, 346
\glxstrgenabbrvfmt 73	\glxstrifindexing 99
\glxstrgeneralpuncIIrules 395	\glxstrifinmark 81, 109–112, 353–355
\glxstrgeneralpuncIrules 395	\glxstrifnextpunc 221, 222
\glxstrgetgrouptitle	\glxstrifperiod 218, 220
..... 145, 436, 439, 441, 453–459, 466	\glxstrifrecordtrigger 173–175
\glxstrgroupfield 166	\GlsXtrIfUnusedOrUndefined 99, 100
\Glsxtrheadfirst 355	\glxstrifwasfirstuse
\glxstrheadfirst 355	.. 82–85, 90–94, 97, 132, 219, 220, 230, 233–240, 255, 257, 280, 282, 297–299, 321–324, 327–329, 331, 341, 343, 347, 349
\Glsxtrheadfirstplural 355	\glxstrinclinkcounter 176
\glxstrheadfirstplural 355	\glxstrindexaliased 98, 99
\Glsxtrheadfull 355	\glxstrindexseealso 61
\glxstrheadfull 355	\glxstrinithyperoutside 79
\Glsxtrheadfullpl 355	\glxstrinitwrgloss 79, 172
\glxstrheadfullpl 355	\glxstrinitwrglossbeforefalse 76
\Glsxtrheadlong 355	\glxstrinitwrglossbeforetrue 76
\glxstrheadlong 355	\Glsxtrinlinefullformat
\Glsxtrheadlongpl 355 228, 230, 244, 245, 253, 256, 258, 260, 262, 264, 271–273, 275, 276, 279, 281, 287, 289–291, 293, 295, 298, 308, 309, 311, 312, 314, 316, 317, 320, 322, 328, 330, 339, 342, 348, 351
\glxstrheadlongpl 355	\glxstrinlinefullformat
\Glsxtrheadname 355 228, 230, 231, 244, 245, 253, 256, 258, 260, 262, 263, 271– 273, 275, 276, 278, 281, 287, 289–291, 293, 295, 298, 308–310, 312, 314, 315, 317, 320, 322, 327, 330, 338, 342, 348, 351
\glxstrheadname 157, 158, 355	\Glsxtrinlinefullplformat 229, 232, 244, 245, 253, 256, 258, 260, 262, 264, 271, 272, 274–276, 279, 281, 288–290, 292, 293, 295, 298, 308, 310–312, 314, 316, 318, 320, 323, 328, 330, 339, 342, 348, 351
\Glsxtrheadplural 355	\glxstrinlinefullplformat
\glxstrheadplural 355 228, 231, 232, 244, 245, 253, 256, 258, 260, 262, 264, 271– 273, 275, 276, 278, 281, 287, 289–291, 293, 295, 298, 308–310, 312, 314, 316, 317, 320, 322, 327, 330, 338, 342, 348, 351
\Glsxtrheadshort 355	\glxstrinsertinsidefalse 246
\glxstrheadshort 354	\GlsXtrInternalLocationHyperlink 27, 147
\Glsxtrheadshortpl 355	\glxstrLatinA 397–402
\glxstrheadshortpl 354	\glxstrLatinAELigature 399, 401, 402
\Glsxtrheadtext 355	
\glxstrheadtext 355	
\glxstrhiernamesep 56, 57	
\glxstrhyperlink 26, 27, 103	
\glxstrhyphensuffix 336, 344	
\glxstridentifyglslike 169	
\glxstrifcounttrigger 119, 120	
\glxstrifcustomdiscardperiod 218	

<code>\glxtrLatinE</code>	397–402	<code>\glxtrMathItalicKappa</code>	407, 411, 412
<code>\glxtrLatinEszettSs</code>	398–400, 402	<code>\glxtrMathItalicLambda</code>	407, 411, 412
<code>\glxtrLatinEszettSz</code>	399, 401	<code>\glxtrMathItalicMu</code>	407, 411, 412
<code>\glxtrLatinEth</code>	398–401	<code>\glxtrMathItalicNu</code>	407, 411, 412
<code>\glxtrLatinH</code>	397–402	<code>\glxtrMathItalicOmega</code>	407, 408, 411, 412
<code>\glxtrLatinI</code>	397–402	<code>\glxtrMathItalicOmicron</code>	407, 411, 412
<code>\glxtrLatinInsularG</code>	401	<code>\glxtrMathItalicPhi</code>	407, 408, 411, 412
<code>\glxtrLatinK</code>	397–402	<code>\glxtrMathItalicPi</code>	407, 411, 412
<code>\glxtrLatinL</code>	397–402	<code>\glxtrMathItalicPsi</code>	407, 408, 411, 412
<code>\glxtrLatinM</code>	397–402	<code>\glxtrMathItalicRho</code>	407, 411, 412
<code>\glxtrLatinN</code>	397–402	<code>\glxtrMathItalicSigma</code>	407, 411, 412
<code>\glxtrLatinO</code>	397–402	<code>\glxtrMathItalicTau</code>	407, 411, 412
<code>\glxtrLatinOELigature</code>	399, 402	<code>\glxtrMathItalicTheta</code>	407, 411, 412
<code>\glxtrLatinP</code>	397–402	<code>\glxtrMathItalicUpsilon</code>	407, 411, 412
<code>\glxtrLatinS</code>	397–402	<code>\glxtrMathItalicXi</code>	407, 411, 412
<code>\glxtrLatinT</code>	397–402	<code>\glxtrMathItalicZeta</code>	407, 411, 412
<code>\glxtrLatinThorn</code>	402	<code>\glxtrmultisupplocation</code>	380
<code>\glxtrLatinX</code>	398–402	<code>\glxtrnamerefink</code>	381
<code>\GlsXtrLocationField</code>	167	<code>\glxtrnewabbrevpresetkeyhook</code>	226
<code>\glxtrlocationhyperlink</code>	147	<code>\glxtrnewnumber</code>	23
<code>\glxtrlocrangefmt</code>	146	<code>\glxtrnewsymbol</code>	23
<code>\GLSxtrlong</code>	22, 93, 362, 363	<code>\glxtrNoGlossaryWarning</code>	16, 25, 148
<code>\Glsxtrlong</code>	22, 92, 363	<code>\GlsXtrNoGlsWarningAutoMake</code>	152
<code>\glxtrlong</code>	21, 22, 92, 362	<code>\GlsXtrNoGlsWarningBuildInfo</code>	152
<code>\glxtrlonghyphen</code>	342	<code>\GlsXtrNoGlsWarningCheckFile</code>	152
<code>\glxtrlonghyphennoshort</code>	338, 339	<code>\GlsXtrNoGlsWarningEmptyMain</code>	152
<code>\glxtrlonghyphenshort</code>	336	<code>\GlsXtrNoGlsWarningEmptyNotMain</code>	152
<code>\glxtrlongnoshortdescname</code>	263, 316, 338	<code>\GlsXtrNoGlsWarningEmptyStart</code>	152
<code>\glxtrlongnoshortname</code>	265, 274, 291, 311, 313, 339	<code>\GlsXtrNoGlsWarningHead</code>	151
<code>\GLSxtrlongpl</code>	22, 94, 363, 364	<code>\GlsXtrNoGlsWarningMisMatch</code>	152
<code>\Glsxtrlongpl</code>	22, 93, 364	<code>\GlsXtrNoGlsWarningNoOut</code>	152
<code>\glxtrlongpl</code>	21, 22, 93, 363	<code>\GlsXtrNoGlsWarningTail</code>	152, 153
<code>\glxtrlongshortdescname</code>	248, 268, 284, 301, 303, 337, 343	<code>\glxtrnopostpunc</code>	139
<code>\glxtrlongshortdescsort</code>	248, 268, 284, 301, 303, 331, 337, 343	<code>\glxtronlydescname</code>	351
<code>\glxtrlongshortdescsort</code>	247, 266, 283, 300, 302, 325, 326, 335, 341	<code>\glxtronlydescsort</code>	351
<code>\glxtrlongshortname</code>	328, 331	<code>\glxtronlyname</code>	350
<code>\glxtrmarkhook</code>	352, 353	<code>\glxtronlysuffix</code>	350
<code>\glxtrMathItalicAlpha</code>	407, 410, 411	<code>\glxtrorg@ifKV@glslink@hyper</code>	74
<code>\glxtrMathItalicBeta</code>	407, 410, 411	<code>\glxtrorglong</code>	187, 224, 248, 338
<code>\glxtrMathItalicChi</code>	407, 408, 411, 412	<code>\glxtrorgshort</code>	187, 224, 248
<code>\glxtrMathItalicDelta</code>	407, 410, 412	<code>\GLSxtrp</code>	109
<code>\glxtrMathItalicEpsilon</code>	407, 410, 412	<code>\Glsxtrp</code>	109
<code>\glxtrMathItalicEta</code>	407, 411, 412	<code>\glxtrp</code>	108, 110
<code>\glxtrMathItalicGamma</code>	407, 410, 411	<code>\glxtrparen</code>	219, 227, 228, 247–251, 253, 256, 258–260, 262–264, 266–276, 278, 279, 281, 283–293, 295, 298, 300–312, 314, 316–318, 320, 322–324, 335–340, 344–346, 351
<code>\glxtrMathItalicIota</code>	407, 411, 412	<code>\glxtrpdfentryfmt</code>	36

<code>\Glsxtrpl</code>	68	<code>\GLSxtrshort</code>	22, 90, 112, 356, 357
<code>\glsxtrpl</code>	68	<code>\Glsxtrshort</code>	21, 22, 90, 357
<code>\glsxtrpostdescription</code> .	139, 199, 217, 434	<code>\glsxtrshort</code>	21, 22, 90, 356
<code>\glsxtrposthyphenlong</code>	347, 349	<code>\glsxtrshortdescname</code> ...	260, 272, 288, 309
<code>\glsxtrposthyphenshort</code>	341, 343	<code>\glsxtrshorthyphen</code>	347, 348
<code>\glsxtrposthyphensubsequent</code>	341, 343, 347, 349	<code>\glsxtrshorthyphenlong</code>	345
<code>\glsxtrpostlink</code>	218	<code>\glsxtrshortlongdescname</code>	250, 270, 286, 305, 307, 345, 348
<code>\glsxtrpostlinkendsentence</code>	218	<code>\glsxtrshortlongdescsort</code>	250, 270, 286, 305, 307, 333, 345, 348
<code>\glsxtrpostlinkhook</code>	218	<code>\glsxtrshortlongname</code>	249, 268, 285, 303, 305, 329, 332, 344, 346
<code>\glsxtrpostlocalreset</code>	115, 117, 125	<code>\glsxtrshortlonguserdescname</code> ..	331, 333
<code>\glsxtrpostlocalunset</code>	115, 117, 125	<code>\glsxtrshortnolongname</code> .	258, 270, 287, 307
<code>\glsxtrpostlongdescription</code>	49	<code>\GLSxtrshortpl</code>	22, 92, 356–358
<code>\glsxtrpostnamehook</code>	204–206, 209	<code>\Glsxtrshortpl</code>	21, 22, 91, 357
<code>\GlsXtrPostNewAbbreviation</code> .	227, 244, 245, 247–249, 251, 252, 254, 255, 257–265, 267–270, 272, 274, 277, 279, 280, 282–288, 291, 294, 296–298, 300–307, 309, 311, 313, 315, 317–319, 321, 323, 325, 326, 328, 329, 331, 332, 334, 336–338, 340, 341, 343–345, 347, 349, 350, 352	<code>\glsxtrshortpl</code>	21, 22, 91, 356
<code>\glsxtrpostreset</code>	115, 117, 125	<code>\glsxtrsmfont</code>	282
<code>\glsxtrpostunset</code>	113, 117, 125	<code>\glsxtrsmsuffix</code>	283, 285, 287, 289, 291, 292, 294, 297
<code>\glsxtrprelocation</code>	424, 426–428, 430, 432, 435, 461	<code>\GlsXtrStandaloneEntryName</code>	158
<code>\glsxtrprotectlinks</code>	102–104	<code>\GlsXtrStandaloneEntryOther</code>	159
<code>\glsxtrprovideaccsuppcmd</code>	190, 191	<code>\GlsXtrStandaloneGlossaryType</code> .	158, 159
<code>\GlsXtrRecordCounter</code>	13	<code>\GlsXtrStandaloneSubEntryItem</code> .	158, 159
<code>\glsxtrrecordtriggervalue</code>	172	<code>\Glsxtrsubsequentfmt</code>	241, 244, 263, 274, 276, 291, 293, 312, 313, 315, 317, 338, 342, 347
<code>\GlsXtrRecordWarning</code>	153	<code>\glsxtrsubsequentfmt</code>	241, 244, 263, 274, 276, 291, 292, 311, 313, 315, 317, 338, 342, 347
<code>\glsxtrregularfont</code>	73, 82	<code>\Glsxtrsubsequentplfmt</code>	241, 244, 263, 274, 276, 291, 293, 312, 313, 315, 317, 338, 342, 347
<code>\glsxtrresourcecount</code>	154	<code>\glsxtrsubsequentplfmt</code>	241, 244, 263, 274, 276, 291, 293, 311, 313, 315, 317, 338, 342, 347
<code>\glsxtrresourcefile</code>	154	<code>\glsxtrsupplocationurl</code> .	147, 148, 380, 382
<code>\glsxtrresourceinit</code>	153	<code>\glsxtrtagfont</code>	217
<code>\glsxtrrestoremarkhook</code>	353	<code>\GLSxtrtitlefirst</code>	370
<code>\glsxtrrestorepostpunc</code>	139	<code>\Glsxtrtitlefirst</code>	354, 355, 370, 371
<code>\glsxtrscfont</code>	266	<code>\glsxtrtitlefirst</code>	354, 355, 370
<code>\glsxtrscsuffix</code>	267, 269, 270, 272, 274, 276, 278, 280	<code>\GLSxtrtitlefirstplural</code>	371
<code>\GlsXtrSetActualChar</code>	213	<code>\Glsxtrtitlefirstplural</code>	354, 355, 371
<code>\glsxtrsetaliasnoindex</code>	16, 17, 98, 99	<code>\glsxtrtitlefirstplural</code>	354, 355, 371
<code>\GlsXtrSetEncapChar</code>	213	<code>\GLSxtrtitlefull</code>	374
<code>\GlsXtrSetEscChar</code>	213	<code>\Glsxtrtitlefull</code>	354–356, 374
<code>\glsxtrsetfieldifexists</code>	42, 43	<code>\glsxtrtitlefull</code>	354, 355, 373, 374
<code>\glsxtrsetglossarylabel</code>	140	<code>\GLSxtrtitlefullpl</code>	375
<code>\GlsXtrSetLevelChar</code>	213	<code>\Glsxtrtitlefullpl</code>	354–356, 375
<code>\glsxtrsetpopts</code>	108		
<code>\glsxtrsetupfulldefs</code>	230–232, 255, 257, 280, 282, 297, 299, 322, 324		

<code>\glxtrtitlefullpl</code>	354–356, 374	<code>\glxtrUpSigma</code>	406, 411, 412
<code>\GLSxtrtitlelong</code>	372	<code>\glxtrUpTau</code>	406, 411, 412
<code>\Glsxtrtitlelong</code>	354, 355, 372	<code>\glxtrUpTheta</code>	405, 406, 411, 412
<code>\glxtrtitlelong</code>	354, 355, 372	<code>\glxtrUpUpsilon</code>	406, 411, 412
<code>\GLSxtrtitlelongpl</code>	373	<code>\glxtrUpXi</code>	406, 411, 412
<code>\Glsxtrtitlelongpl</code>	354, 355, 373	<code>\glxtrUpZeta</code>	405, 406, 411, 412
<code>\glxtrtitlelongpl</code>	354, 355, 372, 373	<code>\GlsXtrUseAbbrStyleFmts</code>	249, 251, 254, 257, 259, 261, 262, 265, 266, 268, 270, 273, 279, 282, 285, 287, 290, 296, 299, 301, 303, 305, 307, 310, 315, 318, 321, 324, 329, 331, 332, 334, 337, 338, 340, 343, 346, 349, 352
<code>\GLSxtrtitlename</code>	368	<code>\GlsXtrUseAbbrStyleSetup</code>	259, 261, 262, 264, 265, 273, 276, 290, 292, 310, 315, 318
<code>\Glsxtrtitlename</code>	354, 355, 368	<code>\glxtrusefield</code>	377
<code>\glxtrtitlename</code>	354, 355, 367	<code>\glxtruserfield</code>	324
<code>\glxtrtitleorpdforheading</code>	28, 29, 157–159, 354, 355	<code>\glxtruserparen</code>	325–334
<code>\GLSxtrtitleplural</code>	370	<code>\glxtrusersuffix</code>	326, 327, 329, 333
<code>\Glsxtrtitleplural</code>	354, 355, 369	<code>\glxtruseseealsoformat</code>	58, 59
<code>\glxtrtitleplural</code>	354, 355, 369	<code>\glxtruseseeformat</code>	55, 58
<code>\Glsxtrtitleshort</code>	354, 355, 367	<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	225, 245
<code>\glxtrtitleshort</code>	354, 355, 366	<code>\GlsXtrWarning</code>	67, 68
<code>\Glsxtrtitleshortpl</code>	354, 355, 367	<code>\glxtrword</code>	224
<code>\glxtrtitleshortpl</code>	354, 355, 366, 367	<code>\glxtrwordsep</code>	224, 335, 337, 340, 344, 346
<code>\GLSxtrtitletext</code>	369	<code>\glxtrwrglossmark</code>	28
<code>\Glsxtrtitletext</code>	354, 355, 368, 369		
<code>\glxtrtitletext</code>	354, 355, 368		
<code>\GlsXtrTotalRecordCount</code>	171		
<code>\glxtrtreechildpredesc</code>	438, 442		
<code>\glxtrtreepredesc</code>	437, 442		
<code>\glxtrtreetopindent</code>	442, 451		
<code>\glxtrundefaction</code>	7, 15–17, 33, 50, 51, 53, 54		
<code>\glxtrundeftag</code>	32, 142, 143		
<code>\GlsXtrUnknownDialectWarning</code>	46		
<code>\glxtrunsrtdo</code>	164, 165		
<code>\glxtrUpAlpha</code>	405, 406, 410, 411		
<code>\glxtrUpBeta</code>	405, 406, 410, 411		
<code>\glxtrUpChi</code>	406, 411, 412		
<code>\glxtrUpDelta</code>	405, 406, 410, 412		
<code>\glxtrUpDigamma</code>	405, 407, 411		
<code>\glxtrUpEpsilon</code>	405, 406, 411, 412		
<code>\glxtrUpEta</code>	405, 406, 411, 412		
<code>\glxtrUpGamma</code>	405, 406, 410, 412		
<code>\glxtrUpIota</code>	406, 411, 412		
<code>\glxtrUpKappa</code>	406, 411, 412		
<code>\glxtrUpLambda</code>	406, 411, 412		
<code>\glxtrUpMu</code>	406, 411, 412		
<code>\glxtrUpNu</code>	406, 411, 412		
<code>\glxtrUpOmega</code>	406, 411, 412		
<code>\glxtrUpOmicron</code>	406, 411, 412		
<code>\glxtrUpPhi</code>	406, 411, 412		
<code>\glxtrUpPi</code>	406, 411, 412		
<code>\glxtrUpPsi</code>	406, 411, 412		
<code>\glxtrUpRho</code>	406, 411, 412		

H

<code>\hangindent</code>	438, 440, 442, 452–454, 458–460, 491, 493, 496
<code>\hbox</code>	423
<code>\hfill</code>	423
<code>\href</code>	102
<code>\hsize</code>	70
<code>\hss</code>	423
<code>\hyperlink</code>	17, 103, 380
<code>\hyperpage</code>	209
<code>\hyperref</code>	102, 148, 378, 383
<code>hyperref package</code>	104, 209, 352, 366, 380

I

<code>\if</code>	66
<code>\if@display</code>	12
<code>\if@glxtr@autoseeindex</code>	30, 55, 60
<code>\if@glxtr@equations</code>	9, 79
<code>\if@glxtr@floats</code>	19
<code>\if@glxtr@format@override</code>	210
<code>\if@glxtr@docdefrestricted</code>	64
<code>\if@glxtr@indexcrossrefs</code>	18, 62
<code>\ifblank</code>	34, 67, 68, 133
<code>\ifbool</code>	33, 49

<code>\ifcase</code> .. 7, 16, 23, 25, 28, 65, 76, 140, 436, 464	<code>\ifglsnogroupskip</code> 424, 427–434, 436, 439, 440, 453, 462, 473, 475–477, 479, 481, 482, 484, 485, 487, 488, 490
<code>\ifcsdef</code> 33, 42, 47, 48, 50–53, 77, 80, 95, 96, 108–112, 122, 137, 144, 145, 158, 166, 169–171, 176, 201–208, 214, 219, 224, 240, 244, 381, 426–433, 494	<code>\ifglslonnumberlist</code> 73
<code>\ifcsstring</code> 33, 197, 243	<code>\ifglslonpostdot</code> 20, 139
<code>\ifcsundef</code> 33, 41, 45–48, 50–52, 65, 69, 71, 104, 117, 122–126, 144, 148, 165, 177, 190, 197, 243, 245, 246, 423, 443, 444, 452, 466, 494	<code>\ifglssanitizesort</code> 136
<code>\ifcsvoid</code> 61, 196	<code>\ifglssubentrycounter</code> 47
<code>\ifdef</code> 16, 23, 28, 30, 33, 36, 41, 45, 47, 48, 53, 54, 58–60, 63, 69, 70, 98, 102, 103, 109–111, 134, 137, 142, 143, 154, 155, 159, 168, 187, 199, 200, 213, 214, 217, 324, 354, 366–375, 378, 383–386, 421, 423–426, 434–437, 439, 441, 453–459, 462, 466, 467	<code>\ifglssused</code> 62, 63, 97, 118, 127, 132, 240, 444–446, 448–450
<code>\ifdefempty</code> 7–9, 39–41, 45, 46, 50–52, 55, 58, 79, 81, 116, 128, 130, 134, 137, 146, 153, 160, 163, 166, 171, 172, 188–190, 215, 224, 240, 386, 387, 463	<code>\ifglsxindy</code> 148, 150
<code>\ifdefequal</code> 44, 45, 64, 101, 128, 130, 152, 166, 168, 207	<code>\ifglsxtr@hyperoutside</code> 80
<code>\ifdefstring</code> 6, 15, 27, 43, 133, 156, 210, 216, 462	<code>\ifglsxtr@printgloss@groups</code> ... 161, 163
<code>\ifdefvoid</code> 55, 60–62, 103, 122, 143, 147, 167, 382	<code>\ifglsxtrininitwrglossbefore</code> ... 76, 80, 172
<code>\ifdim</code> 70, 132, 443–451, 471, 495	<code>\ifglsxtrininsertinside</code> 233–240, 242, 243, 247, 248, 250, 252, 253, 255, 256, 258–264, 267, 269, 271–279, 281, 284–295, 297, 298, 300–304, 306, 308–320, 322, 323, 326–328, 330, 333, 335, 337, 340–344, 346, 348, 350, 351
<code>\IfFileExists</code> 25, 148, 152, 153, 156, 421, 422	<code>\ifHy@hyperindex</code> 209
<code>\ifglossaryexists</code> 54, 162, 376, 377	<code>\ifinlist</code> 114
<code>\ifglshasacronym</code> 21, 152, 376	<code>\ifinlistcs</code> 38, 65
<code>\ifglshasacrshortcuts</code> 23	<code>\ifinner</code> 29
<code>\ifglshasautomake</code> 137, 152, 156	<code>\ifKV@glslink@hyper</code> 74, 78, 80
<code>\ifglshasentrycounter</code> 47	<code>\ifKV@glslink@local</code> 75, 172
<code>\ifglshasentryexists</code> 9, 33, 53, 54, 67, 68, 71, 82, 167, 196, 197, 217, 218, 387	<code>\ifKV@glslink@noindex</code> 8, 9, 13, 36, 99
<code>\ifglshasfieldeq</code> 195	<code>\ifmmode</code> 12, 28, 29
<code>\ifglshasdesc</code> ... 437, 438, 491, 492, 494, 496	<code>\ifnum</code> 18, 40, 41, 63, 64, 118, 119, 126, 127, 143, 154, 167, 172, 383–385, 438, 440, 452, 453, 463, 464, 491, 495, 496
<code>\ifglshasfield</code> 35, 36, 324	<code>\ifst@rred</code> 12
<code>\ifglshaslong</code> 121, 122, 175	<code>\ifstreempty</code> 158, 169, 177, 381, 383–387
<code>\ifglshasparent</code> 158, 159, 161, 164, 444, 446, 447	<code>\ifstrequal</code> 20, 25, 78, 101, 381
<code>\ifglshasshort</code> 56, 57, 73, 82	<code>\ifthenelse</code> 152, 219
<code>\ifglshassymbol</code> 219, 220, 437, 438, 440, 442, 492, 494	<code>\IfTrackedDialectHasMapping</code> 46
<code>\ifglshasindexonlyfirst</code> 99	<code>\IfTrackedLanguageFileExists</code> 376
<code>\ifGlsLongExtraUseTabular</code> 472, 474, 475, 477, 478, 480, 481, 483, 484, 486, 487, 489	<code>\ifundef</code> 17, 27, 39–41, 45, 134, 216, 217, 380, 421, 471
	<code>\ifx</code> 8, 10–12, 15, 31, 32, 59, 69, 71, 79, 130, 133, 134, 138, 141, 146, 147, 154–156, 162, 210, 211, 213, 221–224, 226, 334, 382, 460
	<code>\immediate</code> 63, 118, 127, 148, 149, 156
	<code>\index</code> 210
	<code>\indexspace</code> 424, 435, 462, 466
	<code>\input</code> 375
	<code>\inputencodingname</code> 155
	<code>\InputIfFileExists</code> 63
	<code>\istfilename</code> 134

<code>\item</code>	150, 151, 423–426, 435, 436, 454, 455	<code>\makeatletter</code>	63, 148, 153, 212
J		<code>\makeatother</code>	212
<code>\jobname</code>	63, 148, 150–152, 154, 156	<code>\makebox</code>	423, 452, 453, 495
K		<code>\makefirstuc</code>	216
<code>\key@ifundefined</code>	13, 14, 34, 95, 160, 163, 166	<code>makeglossaries</code>	141
<code>\KV@glslink@hyperfalse</code>	83, 97, 98, 103, 104	<code>\makeglossaries</code>	132, 133, 149–152, 156
<code>\KV@glslink@hypertrue</code>	104	<code>makeindex</code>	497
<code>\KV@glslink@noindexfalse</code>	81, 98	<code>makeindex</code>	16, 132, 497
<code>\KV@glslink@noindextrue</code>	98, 104	<code>\makenoidxglossaries</code>	150
L		<code>\MakeTextUppercase</code>	354
<code>\L</code>	402, 405	<code>\MakeUppercase</code>	354, 355
<code>\l</code>	405	<code>\marginpar</code>	29
<code>\label</code>	27, 140	<code>\markboth</code>	353
<code>\large</code>	493	<code>\markright</code>	353
<code>\LaTeX</code>	150, 151	<code>\mathit</code>	390
<code>\leaders</code>	423	<code>\mathrm</code>	389–391
<code>\leavevmode</code>	50, 78	<code>\maxdimen</code>	70
<code>\let</code>	5, 7–11, 13, 15–17, 19, 21–23, 29–31, 35, 38, 44– 46, 49, 63–66, 69–72, 74–76, 78–94, 96– 102, 104, 105, 108, 113, 114, 116–118, 125, 126, 128, 130–136, 138–145, 153, 154, 156, 159–164, 166–168, 172, 177, 187–189, 201–211, 215–217, 221–226, 230–240, 242–244, 255, 257, 280, 282, 297, 299, 322, 324, 352–356, 377, 378, 381, 382, 421, 435, 442, 444, 455, 463–466	<code>\medskip</code>	152, 165, 494
<code>\letabbreviationstyle</code>	253, 254, 256, 257, 259, 261, 264, 265, 272, 273, 288, 290, 309, 310	<code>\MessageBreak</code>	65, 69, 118, 127, 133, 134, 136, 138, 153, 162, 243
<code>\letcs</code>	34, 39–41, 55, 58, 62, 77, 80, 95, 142–144, 166, 167, 201–209, 214, 467	<code>mfirstuc package</code>	216
<code>\levelchar</code>	213	<code>\mfirstucMakeUppercase</code>	41, 42, 83–89, 91–94, 97, 106, 107, 109, 112, 121, 129, 176, 178–185, 192–194, 204, 205, 208, 231, 232, 234, 236, 238, 240–242
<code>\linewidth</code>	471	<code>\mfu@checkword@arg</code>	216
<code>\listadd</code>	122	<code>\mfu@checkword@do</code>	216
<code>\listbreak</code>	216	<code>\midrule</code>	470, 473, 475, 476, 478, 479, 481, 483, 484, 486, 487, 489
<code>\listcsadd</code>	37	N	
<code>\listcseadd</code>	37, 123	<code>\NeedsTeXFormat</code> ...	5, 376, 422, 461, 468, 491
<code>\listcsgadd</code>	37, 65	<code>\new@atom@glossaryentry</code>	63
<code>\listcsxadd</code>	37, 123	<code>\new@command</code>	378
<code>\listxadd</code>	114	<code>\new@glossaryentry</code>	65, 136
<code>\loadglsentries</code>	65, 149	<code>\new@ifnextchar</code>	35, 96, 121, 169, 173–175, 221, 229–240, 388, 389
<code>\long</code>	49	<code>\newabbr</code>	22
M		<code>\newabbreviation</code>	22
<code>\m@ne</code>	378	<code>\newabbreviationhook</code>	226
<code>\MakeAcronymsAbbreviations</code>	131	<code>\newabbreviationstyle</code> 246, 248–251, 254, 256, 257, 259–270, 272–275, 277, 279–281, 283–288, 290, 292, 294, 296, 298, 299, 301, 303, 305, 307, 309–311, 313–316, 318, 320, 321, 323, 325, 326, 328, 329, 331–333, 335, 336, 338, 339, 341, 343–346, 348, 350, 351
		<code>\newacronym</code>	128, 131

<code>\newacronymhook</code>	129	<code>\newglossarystyle</code>	
<code>\newacronymstyle</code>	130–132	463, 472, 474, 475, 477, 478, 480, 481, 483, 484, 486, 487, 489, 491, 495
<code>\newcommand</code>	5– 15, 17–19, 21–27, 29, 31–58, 60, 62, 63, 66–69, 71–74, 76–78, 81–83, 90, 95, 96, 98–104, 107–117, 119, 121–128, 130– 132, 137–144, 146–151, 153–161, 163– 188, 190–201, 207–224, 227–240, 242– 246, 248–251, 253, 257, 259, 262, 265, 266, 282, 283, 299, 324, 325, 328, 330, 335, 337, 340, 341, 344, 346, 349, 351, 353–383, 385–389, 392–420, 422, 424, 434, 435, 437–444, 451, 452, 461–463, 466–476, 478, 479, 481–487, 489, 492–495	<code>\newif</code>	76, 209, 246, 472
<code>\newcount</code>	140, 154, 168	<code>\newlength</code>	442, 493
<code>\newcounter</code>	27, 176	<code>\newnum</code>	23
<code>\newentry</code>	23	<code>\newrobustcmd</code>	35–39, 42– 44, 58, 59, 64, 77, 82, 96, 97, 108, 109, 121, 139, 144, 157, 159, 165, 169, 170, 173–175, 208, 214, 215, 217, 229–240, 334, 353, 354, 356–366, 387–389, 444–451
<code>\newenvironment</code>	162	<code>\newsym</code>	23
<code>\newglossary</code>	21, 134	<code>\newterm</code>	199
<code>\newglossaryentry</code>	23, 63, 65, 117, 124, 129, 199, 200, 227	<code>\newtoks</code>	223
<code>\newglossaryentry options</code>		<code>\newwrite</code>	63, 134
<code>access</code>	189	<code>\nfss@text</code>	29
<code>alias</code>	19, 54, 58, 60–62	<code>\nobreak</code>	424–426, 435, 492
<code>desc</code>	182, 193	<code>\NoCaseChange</code>	109–112, 158, 356–366
<code>descplural</code>	182, 183, 193	<code>\noexpand</code>	11, 13, 26, 58, 61–63, 78, 129, 148, 149, 153, 161, 164–166, 176, 187, 211, 227, 378, 387, 422, 463–465, 472–486, 488, 489, 496
<code>description</code>	518	<code>\nofiles</code>	151
<code>first</code> 102, 179, 180, 192, 246, 360–362, 370, 497		<code>\noindent</code>	152, 439, 441, 455–459, 492
<code>firstaccess</code>	190	<code>\nopagebreak</code>	435, 462, 466, 491, 496
<code>firstplural</code> ..	180, 192, 246, 361, 362, 371, 497	<code>\nopostdesc</code>	50, 67, 68, 139, 162, 199
<code>group</code>	166, 167	<code>\np@Glsxtr@assign@leveloffset</code> .	140, 141
<code>loclist</code>	37	<code>\ns@GLSxtrfull</code>	231
<code>long</code>	184, 194, 372	<code>\ns@Glsxtrfull</code>	230
<code>longplural</code>	185, 194, 372	<code>\ns@glxtrfull</code>	229
<code>name</code> ...	55, 56, 178, 191, 192, 210, 358, 367	<code>\ns@GLSxtrfullpl</code>	232
<code>plural</code>	179, 192, 246, 359, 360, 369	<code>\ns@Glsxtrfullpl</code>	232
<code>see</code>	18, 19, 30, 54, 55, 60, 62, 65, 66, 134	<code>\ns@glxtrfullpl</code>	231
<code>seealso</code>	19, 55, 58, 60–62, 508	<code>\ns@GLSxtrlong</code>	236
<code>short</code>	56, 183, 194, 223	<code>\ns@Glsxtrlong</code>	235
<code>shortaccess</code>	188	<code>\ns@glxtrlong</code>	235
<code>shortaccessplural</code>	188	<code>\ns@GLSxtrlongpl</code>	240
<code>shortplural</code>	184, 194, 223	<code>\ns@Glsxtrlongpl</code>	239
<code>symbol</code>	181, 193	<code>\ns@glxtrlongpl</code>	238
<code>symbolplural</code>	181, 182, 193	<code>\ns@GLSxtrshort</code>	234
<code>text</code> 56, 102, 178, 179, 192, 246, 248, 359, 368		<code>\ns@Glsxtrshort</code>	233
<code>textaccess</code>	189	<code>\ns@glxtrshort</code>	233
<code>user2</code>	47	<code>\ns@GLSxtrshortpl</code>	238
		<code>\ns@Glsxtrshortpl</code>	237
		<code>\ns@glxtrshortpl</code>	236
		<code>\null</code>	25
		<code>\number</code>	63, 123–126, 153, 167, 169
		<code>\numexpr</code>	123, 124, 126, 167

O	
<code>\O</code>	402, 405
<code>\o</code>	405
<code>\openout</code>	63
<code>\or</code>	7, 16, 17, 23, 24, 28, 65, 76, 436, 465
<code>\org@glossaryentrynumbers</code> ...	71, 138, 162
<code>\org@glossarytitle</code>	138, 162
<code>\org@ifKV@glslink@hyper</code>	78, 80
P	
<code>\p@glshyp@opt</code>	100
<code>\p@glstr@assign@leveloffset</code>	140
package options:	
abbreviations	21
accsupp	24, 178
acronym	21
automake	137, 150, 156
immediate	156
true	156
autoseeindex	30
false	30
counter	
wrglossary	27
debug	
showtargets	103
docdef	17, 64, 65, 117, 124
atom	63
false	64, 65
restricted	18
true	63, 65
docdefs	
restricted	64
equations	9, 79
indexcounter	378
nonumberlist	71
nopostdot	19
false	20
numbers	23
order	
letter	156
postdot	19
prefix	25
record 7, 14, 16, 64, 74, 133, 153, 230–240, 506	
alsoindex	8, 10, 14, 16, 156
hybrid	10, 14, 16
nameref	12, 31, 32, 153, 157
only	8, 151
shortcuts	23
ac	23
all	23
false	23
none	23
true	23
sort	
use	81
style	26
stylemods	26
symbols	23, 199
undefaction	53
error	6
warn	6
xindy	59
<code>\PackageError</code>	6, 13, 26, 30, 64, 65, 69, 77, 95–97, 99, 101, 108, 109, 116–118, 124, 126, 128, 131, 133, 134, 136, 145, 156, 164, 165, 169, 219, 243–246, 423
<code>\PackageWarning</code>	19
<code>\PackageWarningNoLine</code>	19
<code>\pagelistname</code> ...	473, 476, 479, 483, 486, 489
<code>\pageref</code>	27, 47
<code>\par</code>	151, 152, 425, 426, 435, 438–442, 452–454, 456–459, 462, 491–493, 495
<code>\parindent</code>	435, 438, 440, 442, 452–456, 458–460, 463, 491, 493, 496
<code>\parskip</code>	435, 438, 440, 455, 456, 458, 463
<code>\PassOptionsToPackage</code>	5, 25
<code>\pdfbookmark</code>	462, 463
<code>\pdfstringdef</code>	187
<code>\pp@glstr@assign@leveloffset</code>	141
<code>\preglossarypreamble</code>	48
<code>\preto</code>	98, 387
<code>\print@noop@unsrtglossaryunit</code> ...	13, 16
<code>\print@op@unsrtglossaryunit</code>	16, 17
<code>\printabbreviations</code>	21
<code>\printglossaries</code>	15, 133, 150
<code>\printglossary</code>	15, 21, 133, 150, 151, 153
<code>\printglossary options</code>	
nonumberlist	73
type	136
<code>\printnoidxglossaries</code>	150
<code>\printnoidxglossary</code>	135, 150
<code>\printnumbers</code>	23, 200
<code>\printsymbols</code>	23, 199
<code>\printunsrtglossary</code> 151, 153, 160, 376, 377	
<code>\printunsrtglossaryentryprocesshook</code>	161, 163, 164
<code>\printunsrtglossaryhandler</code>	164, 165

<code>\printunsrtglossarypredoglossary</code> ..	<code>\renewcommand</code>
..... 161, 164	6, 7,
<code>\printunsrtglossaryskipentry</code> ..	15–21, 23–28, 30, 31, 33, 48–54, 56, 63–
161, 163	66, 68, 69, 71–76, 78, 95, 97–99, 102–
<code>\printunsrtglossaryunit</code>	104, 108, 115–118, 121, 122, 124–141,
16, 17, 165	143–145, 147–149, 162, 165, 171, 199,
<code>\printunsrtglossaryunitsetup</code>	201–206, 214–218, 228, 229, 244, 245,
165	247–353, 377, 386, 421, 423–436, 438–
<code>\ProcessOptions</code>	441, 452–459, 464, 465, 473–492, 495, 496
423	<code>\renewenvironment</code>
<code>\ProcessOptionsX</code>	424,
28	426–433, 435, 438, 440, 452, 455–459,
<code>\protect</code>	463, 472–478, 480–486, 488, 489, 491, 495
29, 109–112,	<code>\renewglossarystyle</code>
192–194, 224, 227, 228, 246–260, 262– 423–433, 435, 436, 438–441, 452–459
266, 268, 270–280, 282–288, 290–314,	<code>\renewrobustcmd</code>
316–321, 323, 325, 326, 328–332, 334,	81, 103
336–339, 341, 343–345, 347–352, 356–366	<code>\RequireGlossariesExtraLang</code> ...
<code>\protected@csedef</code>	376, 421
42, 43, 144, 443, 444	<code>\RequirePackage</code>
<code>\protected@csxdef</code>	5, 16, 24–26, 28, 422, 461, 468, 491
43, 144, 443, 444	<code>\reserved@a</code>
<code>\protected@eappto</code>	221
13, 50–52, 130, 161, 164, 166, 210	<code>\reserved@b</code>
<code>\protected@edef</code>	221
6, 8, 9, 11, 31, 36, 44, 50–53,	<code>\reserved@d</code>
58, 61, 62, 64, 69, 77, 78, 80, 81, 97, 99,	221, 222
102, 103, 122, 123, 125, 126, 129, 130,	<code>\RestoreAcronyms</code>
134, 135, 140, 141, 143, 144, 146, 158,	128, 131
159, 165, 166, 172, 187–189, 201–208,	<code>\rGLS</code>
210, 214, 222, 226, 244, 382, 387, 446, 447	171
<code>\protected@write</code>	<code>\rGls</code>
11–13, 64, 72, 73,	171
101, 134, 135, 153, 155–157, 168, 387, 466	<code>\rgls</code>
<code>\protected@xdef</code>	171
138, 162, 164	<code>\rGLSformat</code>
<code>\providecommand</code>	174
21, 22, 34, 59, 73,	<code>\rGlsformat</code>
97, 98, 101, 118, 127, 133, 134, 148, 149,	174
155, 168, 376–378, 387, 389–391, 423, 461	<code>\rglsformat</code>
<code>\ProvidesFile</code>	173, 176
375	<code>\rGLSpl</code>
<code>\ProvidesPackage</code> ..	171
5, 376, 422, 461, 468, 491	<code>\rGlspl</code>
	171
	<code>\rGLSplformat</code>
	175
	<code>\rGlsplformat</code>
	174
	<code>\rglsplformat</code>
	173, 176
	<code>\robustify</code>
	377
	<code>\romannumeral</code>
	443, 444, 452, 494
Q	
<code>\quad</code>	
495	
<code>\quotechar</code>	
213	
R	
<code>\raggedright</code>	<code>\s@glstrfmt</code>
428, 429, 432, 433, 464, 470	35
<code>\refstepcounter</code>	<code>\s@glshyp@opt</code>
27	100
<code>\relax</code>	<code>\s@glstr@enabletagging</code>
7, 15,	215
16, 18, 22, 23, 25, 28, 30, 31, 35, 40, 41,	<code>\s@glstrfmt</code>
63–66, 69, 70, 72, 76, 78, 79, 100, 108,	35
118, 119, 126, 127, 134, 135, 138, 140–	<code>\s@GlsXtrIfFieldCmpNum</code>
143, 146, 153, 154, 156, 162, 163, 167,	40
168, 172, 211, 213, 216, 219, 223, 224,	<code>\s@GlsXtrIfFieldEqNum</code>
334, 378, 383–385, 436, 438, 440, 444–	40
454, 458–460, 463–466, 491, 493, 495, 496	<code>\s@GlsXtrIfFieldEqStr</code>
<code>relsize package</code>	43
282	<code>\s@GlsXtrIfFieldEqXpStr</code>
	44
	<code>\s@GlsXtrIfFieldNonZero</code>
	39
	<code>\s@GlsXtrIfFieldValueInCsvList</code>
	38
	<code>\s@glstrifhasfield</code>
	39, 43, 44, 98
	<code>\s@GlsXtrIfXpFieldEqXpStr</code>
	44
S	

<code>\s@GlsXtrStartUnsetBuffering</code>	113	<code>\tablehead</code>	430–433
<code>\s@GlsXtrStopUnsetBuffering</code>	114	<code>\tabletail</code>	430–433
<code>\s@ifglossaryexists</code>	33	<code>\tabularnewline</code>	426–434, 470, 473–490
<code>\s@printunstrtglossary</code>	159, 165	<code>\TeX</code>	150
<code>\seealsoname</code>	58, 59	<code>\texorpdfstring</code>	36, 41, 109–112, 214, 354, 366–375
<code>\seename</code>	55, 58	<code>\textbf</code>	187, 434, 470, 493, 495
<code>\setabbreviationstyle</code>	131, 248, 259	textcase package	352
<code>\SetAcronymLists</code>	130, 131	<code>\textit</code>	187
<code>\setacronymstyle</code>	130–132	<code>\textmd</code>	187
<code>\SetDefaultAcronymDisplayStyle</code>	132	<code>\textrm</code>	187
<code>\setentrycounter</code>	145, 380, 382	<code>\textsc</code>	187, 266
<code>\SetGenericNewAcronym</code>	132	<code>\textsf</code>	187
<code>\setglossarystyle</code>	26, 138, 162, 423, 425, 426, 436, 439, 441, 453–460, 463	<code>\textsl</code>	187
<code>\setkeys</code>	9, 26, 31, 36, 79, 81, 99, 129, 138, 161, 162, 172, 224, 226	<code>\textsmaller</code>	282
<code>\setlength</code>	70, 435, 438, 440, 442, 453, 455, 456, 458, 463, 471, 472, 493, 494	<code>\texttt</code>	29, 149–152, 187
<code>\settowidth</code>	132, 442–452, 471, 494	<code>\the</code>	129, 147, 154, 213, 214, 227, 246–270, 272, 274, 277–280, 282–289, 291, 294, 296–309, 311, 313, 315, 317–323, 325–341, 343–345, 347–352, 382, 494
<code>\setupglossaries</code>	5, 30	<code>\theH</code>	32
<code>\sfcode</code>	20, 219, 434	<code>\theHglentrycounter</code> .	8, 10–12, 79, 81, 172
<code>\small</code>	29, 58	<code>\theHglentrycounter</code> .	8, 10–13, 79, 81, 172
<code>\smallskip</code>	494	<code>\theindex</code>	209
soul package	114	<code>\thewrglossary</code>	27
<code>\space</code>	6, 13, 15, 59, 65, 66, 69, 90, 99, 116–118, 124, 126–128, 131–134, 136, 138, 149, 152, 153, 156, 162, 164, 165, 169, 219, 223, 228, 248, 423, 424, 434, 437, 438, 440, 442, 452, 461, 492, 494, 495	<code>\this@dialect</code>	376, 421
<code>\spacefactor</code>	20, 219, 434	<code>\toks@</code>	147, 213, 214, 382
<code>\stepcounter</code>	177	<code>\toprule</code>	470, 473, 475, 476, 478, 479, 481, 482, 484, 486, 487, 489
<code>\string</code>	6, 11–13, 15, 32, 59, 64–66, 69, 72, 73, 80, 90, 95, 96, 99, 101, 108, 109, 116–118, 124, 126–128, 131, 133–136, 138, 148–153, 155–157, 162, 164, 165, 168, 169, 203, 204, 206, 208, 210, 219, 377, 378, 387, 392–421, 466	<code>\TrackedDialectClosestSubMatch</code>	45
<code>\strut</code>	423–434, 440, 469	tracklang package	45, 155, 391
<code>\sty</code>	133	<code>\TrackLangGetDefaultScript</code>	421
<code>\subglossentry</code>	138, 162, 167, 423–434, 436, 438, 440, 452, 464, 473, 475–477, 479, 480, 482, 484, 485, 487, 488, 490, 491, 496	<code>\TrackLangIfHasDefaultScript</code>	421
<code>\subitem</code>	435, 436	<code>\TrackLangRequireDialectPrefix</code>	421
<code>\subsubitem</code>	435, 436	<code>\triangleright</code>	58
<code>\symbolname</code>	471, 478, 479, 481, 482, 484, 486, 487, 489	U	
T		<code>\u</code>	377
<code>\tabcolsep</code>	471, 472	<code>\undef</code>	16, 17, 63, 215
		<code>\underline</code>	217
		<code>\unskip</code>	50, 63, 423
		upgreek package	390
		<code>\usepackage</code>	151, 152
		W	
		<code>\warn@nomakeglossaries</code>	135
		<code>\warn@noprntglossary</code> ..	133, 135, 138, 163
		wrglossary (counter)	26, 27
		<code>\write</code>	59, 118, 127, 134, 148, 149, 156

X	
\xcapitalisewords	201
\xdef	378
\xifinlist	123
\xifinlistcs	38
xindy	497
xindy	16, 132, 497
xkeyval package	5
\XKV@checkchoice	73
\XKV@plfalse	73
\XKV@resa	73
\XKV@sttrue	73